BAYESIAN METHODS FOR UNCERTAINTY QUANTIFICATION

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Ilias Bilionis

August 2013

BAYESIAN METHODS FOR UNCERTAINTY QUANTIFICATION

Ilias Bilionis, Ph.D.

Cornell University 2013

Computer codes simulating physical systems usually have responses that consist of a set of distinct outputs (e.g., velocity and pressure) that evolve also in space and time and depend on many unknown input parameters (e.g., physical constants, initial/boundary conditions, etc.). Furthermore, essential engineering procedures such as uncertainty quantification, inverse problems or design are notoriously difficult to carry out mostly due to the limited simulations available. The aim of this work is to introduce a fully Bayesian approach for treating these problems which accounts for the uncertainty induced by the infinite number of observations.

BIOGRAPHICAL SKETCH

Ilias Bilionis was born in Athens, Greece on May 3 1985. He graduated from high school on June 2003. Then, he studied Applied Mathematics at the National Technical University of Athens majoring in Mathematical Analysis and Statistics. He obtained his Diplomma on June 2008. On August 2008, he enrolled to the Ph.D. program at the Center for Applied Mathematics at Cornell University. On August 2010, he joined the group of Professor Nicholas Zabaras (Material Process Design and Control Laboratory, Sibley School of Mechanical and Aerospace Engineering).

Στην Κ.

Αφιερωμένο, στη γυναίκα.

ITHACA

As you set out for Ithaca,
hope that your journey is a long one,
full of adventure, full of discovery.
Laistrygonians and Cyclopes,
angry Poseidon – do not be afraid of them:
as long as you keep your thoughts raised high,
as long as a rare sensation
touches your spirit and your body.
Laistrygonians and Cyclopes,
wild Poseidon – you won’t encounter them
unless you bring them along inside your soul,
unless your soul sets them up in front of you.
Hope that your journey is a long one.
May there be many summer mornings when,
with what pleasure, what joy
you come into harbors you’re seeing for the first time;
may you stop at Phoenician trading stations,
to buy fine things,
mother of pearl and coral, amber and ebony,
sensual perfume of every kind-
as many sensual perfumes as you can;
and may you visit many Egyptian cities
to learn and learn again from those who know.
Keep Ithaca always in your mind.
Arriving there is what you’re destined for.
But don’t hurry the journey at all.
Better if it lasts for years,
so that you are old by the time you reach the island,
wealthy with all you’ve gained on the way,
not expecting Ithaca to make you rich.
Ithaca gave you the marvelous journey.

Without her you would have not set out.

She has nothing left to give you now.

And if you find her poor,

Ithaca won't have fooled you.

Wise as you will have become, so full of experience,

you'll have understood by then what these Ithacas mean.

C. P. Cavafis, Alexandria, Egypt, 1893

ACKNOWLEDGEMENTS

I would like to start by thanking my advisor, Professor N. Zabaras, for his unconditional support all these years and for bringing me in contact with a great variaty of important scientific topics. Special thanks goes to Professor P. S. Koutsourelakis for teaching me how to think like a Bayesian. I would like to thank my Ph.D. committee, Professors A. Vladimirsky, D. Bindel and G. Samorodnitsky for their support. Their comments on my presentation skills where invaluable. I thank my parents for teaching me the value of education and for never denying to buy me a book! Finally, I am greateful to Katerina for her patience during my years at Cornell, her unconditional support and love.

CONTENTS

CHAPTER 1

INTRODUCTION

It is very common for a research group or a company to spend years of development of sophisticated software in order to simulate realistically important physical phenomena. However, carrying out tasks like uncertainty quantification, model calibration or design using the full-fledged model is -in all but the simplest cases- a daunting task, since a single simulation might take days or even weeks to complete, even with state-of-the-art modern computing systems. One, then, has to resort to computationally inexpensive surrogates of the computer code. The idea is to run the solver on a small, well-selected set of inputs and then use these data to learn the response surface. The surrogate surface may be subsequently used to carry out any of the computationally intensive engineering tasks.

The engineering community and, in particular, the researchers in uncertainty quantification, have been making extensive use of surrogates, even though most times it is not explicitly stated. One example is the so-called *stochastic collocation* (SC) method (see [3] for a classic illustration) in which the response is modeled using a generalized Polynomial Chaos (gPC) basis [99] whose coefficients are approximated via a collocation scheme based on a tensor product rule of one-dimensional Gauss quadrature points. Of course, such approaches scale extremely badly with the number of input dimensions since the number of required collocation points explodes quite rapidly. A partial remedy of the situation can be found by using sparse grids (SG) based on the Smolyak algorithm [85], which have a weaker dependence on the dimensionality of the problem (see [95, 96, 69] and the adaptive version developed by our group [59]).

Despite the rigorous convergence results of all these methods, their applicability to the situation of very limited observations is questionable. In that case, a statistical approach seems more suitable.

To the best of our knowledge, the first attempt of the statistics community to build a computer surrogate starts with the seminal papers of Currin et al. [22] and -independently- Sacks et al. [81], both making use of Gaussian processes. In the same spirit is also the subsequent paper by Currin et al. [23] and the work of Welch et al. [93]. One of the first applications to uncertainty quantification can be found in O'Hagan et al. [74] and Oakley and O'Hagan [71]. The problem of model calibration is considered in [52] and in [50]. References [41] and [4] model non-stationary responses, while [19] and [50] (in quite different ways) attempt to capture correlations between multiple outputs. Following these trends, we will consider a Bayesian approach to the problem of uncertainty quantification.

Despite the simplistic nature of the surrogate idea, there are still many hidden obstacles. Firstly, the question about the choice of the design of the inputs on which the full model is to be evaluated arises. It is generally admitted that a good starting point is a Latin hyper-cube design [64], because of its great coverage properties. However, it is more than obvious, that this choice should be influenced by the task in which the surrogate will be used. For example, in uncertainty quantification, it makes sense to bias the design using the probability density of the inputs [4] so that highly probable regions are adequately explored. Furthermore, it also pays off to consider a sequential design that depends on what is already known about the surface. Such a procedure, known as active learning, is particularly suitable for Bayesian surrogates since one may use the predictive variance as an indicative measure of the informational content of particular points in the input space [62]. Secondly, computer codes solving partial differential equations usually have responses that are multi-output functions of space and/or time. One can hope, that explicitly modeling this fact may squeeze more information out of the observations. Finally, it is essential to be able to say something about the epistemic uncertainty induced by the limited number of observations and, in particular, about its effect on the task for which the surrogate is constructed.

This thesis is the result of three papers I published on uncertainty quantification during my PhD, [4, 5] and most recently [6]. Even though the underlying problem remains the same in all three papers, each one is addressing only some of the underlying intricacies. The common characteristic of all the developed methodologies is that they are Bayesian. The purpose of the this introductory chapter is to clearly define the uncertainty quantification problem and discuss the various difficulties that arise.

1.1 The uncertainty quantification problem

A computer code may be thought as a function $f : \mathcal{X} \rightarrow \mathcal{Y}$, such that if an input $x \in \mathcal{X}$ is supplied to it, it responds with $y = f(x) \in \mathcal{Y}$. The response function $f(x)$ is not known analytically. It can however be evaluated at will for a given cost. Therefore, we can think of its evaluation as an experiment, a computer experiment.

In uncertainty quantification applications, we assume that the input is associated with a probability space $(\mathcal{X}, \mathcal{F}, P)$, where $\mathcal{F}$ is a σ-algebra and P a probability measure. In most applications of interest, $\mathcal{X}$ is a subset of $\mathbb{R}^k$ and $\mathcal{F}$ will be identified with a subset of the Borel σ-algebra. For simplicity, we will also assume that P is absolutely continuous, i.e., there exists a density function $p(x)$ s.t.

$$p(A) = \int_A p(x)dx,$$

for all $A \in \mathcal{F}$.

The uncertainty of the inputs is either *aleatoric*, i.e. a random process affecting the response, or *epistemic*, i.e. lack of knowledge about the true value of x. Of course, any combination of the two is also allowed. Aleatoric uncertainty is irreducible, while epistemic uncertainty can be reduced by refining the way x is measured. No distinction will be made between aleatoric and epistemic uncertainty in this work.

The problem of uncertainty quantification consists of identifying the input-induced response probability space $(\mathcal{Y}, \mathcal{G}, Q)$. Formally:

$$\begin{aligned}
\mathcal{Y} &= \{y : f(x) = y, x \in \mathcal{X}\}, \\
\mathcal{G} &= \{B : f(A) = B, A \in \mathcal{F}\}, \\
Q(B) &= P\left(f^{-1}(B)\right), \text{ for } B \in \mathcal{G}.
\end{aligned}$$

In case Q is also absolutely continuous, we may also be interested in its density $q(y)$. Identifying the full probability space $(\mathcal{Y}, \mathcal{G}, Q)$ is almost always a futile task. Therefore, we are usually interested in capturing finite order statistics of the response. Typical examples are the *mean*:

$$\mathbf{E}[y] := \int_{\mathcal{Y}} y dQ(y) = \int_{\mathcal{X}} f(x) dP(x), \tag{1.1}$$

and the *variance*

$$\mathbf{V}[y] := \mathbf{E}\left[(y - \mathbf{E}[y])^2\right]. \tag{1.2}$$

1.2 The Monte Carlo approach to uncertainty quantification

The Monte Carlo (MC) approach to UQ is a straight-forward intuitive procedure. One draws n random samples from the input probability space $\left\{x^{(i)}\right\}_{i=1}^{n}$, evaluates the corresponding responses $\left\{y^{(i)} = f\left(x^{(i)}\right)\right\}_{i=1}^{n}$ and estimates, e.g., the mean (see Equation 1.1) by

$$\mathbf{E}\left[y\right] \approx \frac{1}{n}\sum_{i=1}^{n} y^{(i)} = \frac{1}{n}\sum_{i=1}^{n} f\left(x^{(i)}\right). \tag{1.3}$$

A similar estimator can be easily constructed also for the variance (see Equation 1.2), while kernel density methods [84] can be employed to approximate the density of y.

MC's wide acceptance is due to the fact that it can uncover the complete statistics of the solution, while having a convergence rate that is (remarkably) independent of the input dimension ($(1/\sqrt{n})$, where n is the number of random samples used [56]). Nevertheless, it quickly becomes inefficient in high dimensional and computationally intensive problems, where only a few samples can be observed. Such difficulties have been (partially) alleviated by improved sampling techniques such as Latin hypercube sampling [51] and multilevel MC [35, 36].

The difficulties with MC are more pronounced when one only has a limited set of observations, i.e. small n. This is usually the case we encounter when $f(x)$ is very computationally intensive, e.g., it might take a couple of hours to a day, or even a week for a single evaluation. The situation was first described in a seminal paper by O'Hagan, suitably termed "Monte Carlo is Fundamentally Unsound" [72]. There are basically two important objections to MC raised by O'Hagan, which I will briefly discuss.

A very important variant of MC is importance sampling [56]. Instead of sampling $\left\{x^{(i)}\right\}_{i=1}^{n}$ from the input density $p(x)$, one samples from an other density $q(x)$ and uses the following estimator for Equation 1.3:

$$\mathbf{E}[y] \approx \frac{1}{n}\sum_{i=1}^{n}\frac{f\left(x^{(i)}\right)p\left(x^{(i)}\right)}{q\left(x^{(i)}\right)}.$$

Notice that the estimator does not only depend on the observed values of $f(x)$. It also depends on the sampling density which is completely arbitrary and carries no information about $f(x)$ or each expectation value. In statistical terms, we conclude that the MC estimator violates the Likelihood Principle. The effect of this is diminished if $q(x)$ is built using prior knowledge about $f(x)$ and, of course, if the number of observations n increases. However, at finite n the effect can be huge and unpredictable.

The second objection rests on the fact that the MC estimator of Equation 1.3 does not make use of any information contained in the input points $x^{(i)}$. Notice that it only uses the function values $f\left(x^{(i)}\right)$. To shed some light into the problem, assume that $n = 3$ and that $x^{(2)} = x^{(3)}$ (or that they are close together and that the function $f(x)$ is very smooth). Then $f\left(x^{(2)}\right) = f\left(x^{(3)}\right)$, and there is clearly nothing to be learned by including $x^{(3)}$ in the estimation. However, Equation 1.3 gives:

$$\mathbf{E}[y] \approx= \frac{1}{3}\left(f\left(x^{(1)}\right) + 2f\left(x^{(2)}\right)\right),$$

which is clearly unreasonable.

1.3 The Bayesian approach to uncertainty quantification

The common feature of the following chapters is that they develop UQ methodologies that are fully Bayesian in nature. In this small section, I will simply highlight the core idea behind this line of thinking without making explicit use of a specific model. In particular, we will provide the Bayesian answer to the following question: Given a set of observations:

$$\mathcal{D} = \left\{\left(x^{(i)}, f\left(x^{(i)}\right)\right)\right\}, \tag{1.4}$$

what is the best estimate of Equation 1.1 and/or Equation 1.2? The same ideas, may of course be generalized to more exotic statistics in a straightforward manner. The seeds of these ideas can be traced back to the works of O'Hagan [73], and Rasmussen and Ghahramani [78].

For simplicity, we concentrate on the expectation of $f(x)$:

$$I = \int f(x)p(x)dx.$$

We would like to estimate it using $\mathcal{D}$. The idea is to construct a Bayesian model for the function $f(x)$ and subsequently use it to estimate I. Towards this goal, let $p(\hat{f}(\cdot))$ denote a prior probability density for $f(\cdot)$. You may think of it as the definition of a random field over $\mathcal{X}$. It should encode what we believe about the function $f(\cdot)$ before we see any data, e.g., is it continuous, does it have smooth first derivatives, is it positive only or negative only, is it periodic? In practice, $p\left(\hat{f}(\cdot)\right)$ is defined through some kind of discretization introducing a finite set of parameters (e.g., $f(\cdot)$ could be expanded in a set of basis functions and the random field would be induced through a probability density on the set of coefficients). Here, we will simply abuse the notation for the sake of simplicity. Specific examples of what $p\left(\hat{f}(\cdot)\right)$ is, are given in the following chapters. The bottom line is that $p\left(\hat{f}(\cdot)\right)$ encodes our prior belief about $f(\cdot)$.

We immediately see, that a prior belief about $f(\cdot)$ automatically induces a prior belief about the value of I. To be specific, define

$$I\left[\hat{f}(\cdot)\right] := \int \hat{f}(x)p(x)dx.$$

Then, before we see any data, we have a probability density over the values of I, namely:

$$p(I) = \int \delta\left(I\left[\hat{f}(\cdot)\right] - I\right) p\left(\hat{f}(\cdot)\right) d\hat{f}(\cdot),$$

where $\delta(\cdot)$ is Dirac's delta function.

Suppose now that we observe $\mathcal{D}$ (see Equation 1.4). Bayes rule, dictates that we should update our knowledge about $f(\cdot)$ as follows:

$$p\left(\hat{f}(\cdot)|\mathcal{D}\right) \propto p\left(\mathcal{D}|\hat{f}(\cdot)\right) p\left(\hat{f}(\cdot)\right), \tag{1.5}$$

where $p(\mathcal{D}|\hat{f}(\cdot))$ is the likelihood of the data (which models any noise that might be present in the measurements). This corresponds to a *posterior* random field modeling our state of knowledge about the function $f(\cdot)$ after seeing $\mathcal{D}$. This, in turn, induces a posterior probability density on the values of I, that is:

$$p(I|\mathcal{D}) = \int \delta\left(I\left[\hat{f}(\cdot)\right] - I\right) p(\hat{f}(\cdot)|\mathcal{D})d\hat{f}(\cdot). \tag{1.6}$$

In the chapters that follow 1) I propose several ways of modeling $f(\cdot)$ based on a finite set of observations addressing different difficulties that one might encounter in practice, 2) I propose techniques that adaptively select the observations $\mathcal{D}$ to be made, targeted at reducing the uncertainty in Equation 1.6, and 3) I propose analytic and/or numerical techniques to approximate Equation 1.6. I have demonstrated with several numerical examples, that the Bayesian approach can outperform significantly MC-based or even collocation based techniques.

CHAPTER 2

MULTI-OUTPUT GAUSSIAN PROCESS REGRESSION

2.1 Introduction

Uncertainty Quantification (UQ) is a field of great importance in practically all engineering tasks. Physical models require as input certain parameters such as physical constants, equations of state, geometric specification of objects, boundary conditions, initial conditions and so on. In general, exact knowledge of these quantities is impossible either due to measurement errors or because they are truly random. As a consequence, both the input parameters as well as the physical responses have to be modeled as random variables. The goal of UQ is to study the propagation of uncertainty from the parameter space to the response space. The most celebrated method for the solution of UQ problems is the Monte Carlo (MC) method. MC's wide acceptance is due to the fact that it can uncover the complete statistics of the solution, while having a convergence rate that is (remarkably) independent of the input dimension. Nevertheless, it quickly becomes inefficient in high dimensional and computationally intensive problems, where only a few samples can be observed. Such difficulties have been (partially) alleviated by improved sampling techniques such as Latin hypercube sampling [51] and multilevel MC [35, 36].

Another approach to UQ is the so called *spectral finite element method* [33]. It involves the projection of the response on a space spanned by orthogonal polynomials of the random variables and the solution of a system of coupled deterministic equations involving the coefficients of these polynomials. The scheme was originally developed for Gaussian random variables which correspond to Hermite polynomials (polynomial chaos (PC)). It was later generalized to include other types of random variables (generalized PC (gPC)) [99]. Due to the global support of the polynomials used, gPC suffers from the well-known Gibbs phenomenon in the presence of discontinuities in the random space. The multi-element generalized polynomial chaos (ME-gPC) method [91, 92] was introduced in order to address exactly this issue. The idea of the multi-element (ME) approach is to decompose the stochastic space in disjoint elements and then employ gPC on each element. However, the coupled nature of the equations that determine the coefficients of the polynomials make the application of the method to high input dimensions extremely difficult (*curse of dimensionality*).

Throughout the chapter, we assume that we have at hand a well-established computer code that emulates the physical system. In fact, we will investigate the propagation of uncertainty from the input of the computer code to the output, by learning the response surface using well selected observations. Any modeling or discretization error will be ignored in this study. The so called *stochastic collocation* methods have been designed to deal with this situation. The response is represented as an interpolative polynomial of the random input constructed by calls to the computer code at specific input points. However, the construction of the set of interpolation points is non-trivial, especially in high-dimensional settings. In [3], a Galerkin based approximation was introduced in conjunction with a collocation scheme based on a tensor product rule using one-dimensional Gauss quadrature points. Despite its appeal, the method scales badly with the number of random input dimensions. Alternatively, sparse grids (SG) based on the Smolyak algorithm [85] have a weaker dependence on the input dimensionality. In [98, 96, 70], the Smolyak algorithm is employed to build sparse grid interpolants in high-dimensional input spaces based on Lagrange interpolation polynomials. Similarly to gPC, such methods also fail to capture local features of the response. From the above discussion, it is apparent that discontinuities in the stochastic space must be dealt with using a basis with local support. In [59], the authors developed an adaptive version of SG collocation (SGC) based on localized hat functions called Adaptive SGC (ASGC). ASGC is able to refine the sparse grid only in important regions of the stochastic space, e.g. near a discontinuity. Nevertheless, the piecewise linear nature of the scheme performs poorly when only a few samples are used while adverse functions can trick the adaptive algorithm into stopping without converging.

Highly sophisticated computer codes modeling real-life phenomena (like weather, ocean waves, earthquakes, etc.) might take hours or even days to complete a single run in massively parallel systems. Therefore, we are necessarily limited to observing only a few realizations. Motivated by this situation, we would like to consider the problems of (1) selecting the most informative observations and (2) quantifying the uncertainty in the prediction of the statistics. From the above mentioned methods, ASGC addresses only problem (1), albeit in an ad hoc manner. In order to deal with (1) and (2) in a principled, information theoretic way, a Bayesian framework is necessary. To this end, we choose to investigate the performance of the Gaussian process (GP) model. The GP model has been used in computer experiments in the pioneering work of Sacks [81] (for a more recent review see the book [82]). GP is particularly interesting, since it provides an analytically tractable Bayesian framework where prior information about the response surface can be encoded in the covariance function, and the uncertainty about the prediction is easily quantified. It is exactly this uncertainty in the prediction that can be exploited in order to select the observations to be made (see [62]), as well as to quantify the uncertainty in the statistics. One of the drawbacks of GP inference is that it scales as the cube of the number of observations, making the treatment of large data sets computationally demanding. Furthermore, the most common covariance functions used in practice are stationary. The effect of the stationarity assumption is that it makes non-stationary responses and localized features (such as discontinuities) a priori highly improbable, resulting in an excessive number of samples being required in order to uncover them. A successful effort to deal with these difficulties has been carried out in [41]. Based on the partitioning ideas of the Bayesian CART model [16, 17], a treed GP model was introduced. By making the GP local to each leaf of the tree, the model is able to process many more samples. Additionally, anisotropy is captured by considering the true response as being the result of many local stationary (albeit different) models. More recently, in [86] a novel tree model was introduced using Sequential Monte Carlo inference as opposed to MCMC of the classical approaches. The latter is a promising step towards computationally tractable fully Bayesian trees.

In this work, we present a novel non-intrusive UQ framework based on a treed multi-output Gaussian process (GP). It operates in two stages: (a) the construction of a surrogate model for the physical response and (b) the interrogation of this surrogate for the statistics. The building block of the surrogate is a Multi-output Gaussian Process (MGP) introduced in Section 2.2.1. Information gathered from the MGP is used to discover important directions of the stochastic space and decompose it in *stochastic elements* (i.e. new leaves of the tree) (Section 2.2.4). Each stochastic element is, in turn, sampled using Sequential Experimental Design (SED) techniques (Section 2.2.5) and subsequently modeled using a new MGP. This defines an iterative procedure that gradually resolves local features and discontinuities. The final result is a piecewise surrogate in the spirit of the Multi-element Method (ME) [91]. Despite being a treed GP, our model differs from the model in [41] in several aspects: 1) the tree building process is inspired from the ME method rather than Bayesian CART (non-probabilistic tree), 2) we explicitly derive point estimates of the missing hyper-parameters by maximizing the marginal likelihood instead of averaging (fast predictions), 3) we treat the multiple outputs of the response in a unified way (faster training). Furthermore, our model is built specifically to deal with UQ tasks, in that the input probability distribution plays an important role in the tree construction. Finally, the resulting surrogate can be used to obtain semi-analytic estimates of the moments of any order as well as error bars (Sections 3.2.4 and 3.2.5).

2.2 Methodology

Let $\mathbf{X} \subset \mathbb{R}^K$ for some $K \geq 1$ represent the stochastic input space, a (potentially infinite) rectangle of $\mathbb{R}^K$, i.e. $\mathbf{X} = \times_{k=1}^{K}[a_k, b_k], -\infty \leq a_k < b_k \leq \infty$. We will assume that there is a probability density $p(\mathbf{x})$ defined for all $\mathbf{x} \in \mathbf{X}$ such that:

$$p(\mathbf{x}) = \prod_{k=1}^{K} p_k(x_k), \tag{2.1}$$

where $p_k(x_k)$ is the probability density pertaining to the k-th dimension. That is, the components of $\mathbf{x}$ are independent random variables. This assumption is very common in UQ settings and can be made to hold by a transformation of the input space. We now consider the multi-output function $\mathbf{f} : \mathbf{X} \to \mathbb{R}^M$ representing the result of a computer code (deterministic solver) modeling a physical system, i.e. at a given input point $\mathbf{x} \in \mathbf{X}$ the response of the system is $\mathbf{f}(\mathbf{x})$. We will write

$$\mathbf{f}(\cdot) = (f_1(\cdot), \dots, f_M(\cdot)),$$

and refer to $f_r(\cdot)$ as the r-th output of the response function, $r = 1, \dots, M$. In this work, we will identify $\mathbf{f}(\cdot)$ as the true response of an underlying physical system and we will ignore any modeling errors. The input probability distribution induces a probability distribution on the output. The UQ problem involves the calculation of the statistics of the output $\mathbf{y} = \mathbf{f}(\mathbf{x})$. Quantities of interest are the *moments* $\mathbf{m}^q = (m_1^q, \dots, m_M^q)$, defined for $q \geq 1$ and $r = 1, \dots, M$ by:

$$m_r^q := \int_{\mathbf{X}} f_r^q(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}, \tag{2.2}$$

as well as functions of them. In particular, the *mean* $\mathbf{m} = (m_1, \dots, m_M)$:

$$m_r := m_r^1 = \int_{\mathbf{X}} f_r(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}, \tag{2.3}$$

and the *variance* $\mathbf{v} = (v_1, \dots, v_M)$:

$$v_r := \int_{\mathbf{X}} \left(f_r(\mathbf{x}) - m_r\right)^2 p(\mathbf{x}) d\mathbf{x} = m_r^2 - (m_r^1)^2. \tag{2.4}$$

The statistics will be calculated by interrogating a surrogate of $\mathbf{f} : \mathbf{X} \to \mathbb{R}^M$. This will be put together from local surrogates defined over elements of the stochastic space $\mathbf{X}^i \subset \mathbf{X}$ such that:

$$\mathbf{X} = \cup_{i=1}^{I} \mathbf{X}^i \text{ and } \operatorname{int}(\mathbf{X}^i) \cap \operatorname{int}(\mathbf{X}^j) = \emptyset, \forall i, j \in I, i \neq j, \tag{2.5}$$

where $\operatorname{int}(\mathbf{X}^i)$ denotes the interior of the set $\mathbf{X}^i$ under the usual Euclidean metric of $\mathbb{R}^K$. The response surface is correspondingly decomposed as:

$$\mathbf{f}(\mathbf{x}) := \sum_{i=1}^{I} \mathbf{f}^i(\mathbf{x}) 1_{\mathbf{X}^i}(\mathbf{x}), \tag{2.6}$$

where $1_{\mathbf{X}^i}(\mathbf{x})$ is the indicator function of $\mathbf{X}^i$, given by:

$$1_{\mathbf{X}^i}(x) = \begin{cases} 1 & \text{if } \mathbf{x} \in \mathbf{X}^i, \\ 0 & \text{otherwise} \end{cases},$$

and $\mathbf{f}^i(\cdot)$ is just the restriction of $\mathbf{f}(\cdot)$ on $\mathbf{X}^i$. The local surrogates will be identified as Multi-Output Gaussian Processes (MGP) defined over the stochastic element $\mathbf{X}^i$. These MGPs will be trained by observing $\mathbf{f}^i(\cdot)$. The predictive mean of the MGPs will be used to derive semi-analytic estimates of all moments $\mathbf{m}^q$. An addendum of the Bayesian treatment, is the ability to provide error bars for the point estimates of the moments. This feature is absent from most current UQ methods.

Our aim is to create a surrogate by making as few calls to the computer program as possible. This is achieved by an interplay of adaptively decomposing the domain (Tree Construction) and selecting which observations to make within each element (Experimental Design). These decisions should be biased by the underlying input probability density $p(\mathbf{x})$ and the observed variability of the responses.

In the sections that follow, we introduce the constituent parts of our framework. Despite the fact that the method is applicable to any distribution $p(\mathbf{x})$ over $\mathbf{X}$, all numerical examples will be conducted on a compact $\mathbf{X}$ (a_k and b_k are finite) using the uniform distribution. This is mainly due to the fact that the implementation of the framework is considerably easier for this case. We plan to investigate and report the dependence of the results on $p(\mathbf{x})$ in a future work.

2.2.1 Multi-output Gaussian Process Regression

We turn our focus to a single element of the stochastic space $\mathbf{X}^i \subset \mathbf{X}$ and discuss the construction of a local surrogate model based on some already observed data. The choice of the elements is the subject of Section 2.2.4 and how the observations are selected is investigated in Section 2.2.5. All quantities introduced herein are local to the element $\mathbf{X}^i$. However, in order to avoid having an unnecessarily complicated notation, we do not explicitly show this dependence.

We assume that we have observed a fixed number $N \geq 1$ of data points

$$\mathcal{D} := \left\{(\mathbf{x}^{(n)}, \mathbf{y}^{(n)}\right\}_{n=1}^{N}, \tag{2.7}$$

where, $\mathbf{y}^{(n)} = \mathbf{f}(\mathbf{x}^{(n)})$ is the result of the computer program with input $\mathbf{x}^{(n)}$. We will fit these data to a Gaussian Process (GP) model [61, 77], a procedure known as GP Regression (GPR). Our primary concern in this section is to extend GPR to the multi-output case. The naive approach would be to model its output dimension independently. However, since the various outputs of the response function are highly correlated, this strategy will incur some loss of information. Furthermore, training a GP on N data points involves the computation of the Cholesky decomposition of an $N \times N$ symmetric positive-definite matrix, an operation that scales as $O(N^3)$. If the M output dimensions were to be modeled independently, then the total training cost would be $O(MN^3)$ making the method inappropriate for most UQ tasks. Several techniques exist that model the correlation between outputs: e.g. 'co-kriging' (Section 3.2.3 in [21]) or introducing latent (hidden) outputs [11, 87, 66]. Unfortunately, these models are still fairly complicated and computationally demanding. In [50], a principal components analysis (PCA) was performed on the output space and then the PCA coefficients of the simulations were modeled using independent GPs. This approach has been proven efficient in dealing with high-dimensional output settings, since it automatically takes care of output correlations. However, it introduces an additional error arising from the finite truncation of the PCA decomposition of the output field. Furthermore, it is not clear how the approach can be used in a SED setting, in which simulations arrive one by one, as well as how it performs when discontinuities are present in the stochastic space. A very recent, theoretically sound way of modeling multiple outputs was developed in [19]. In this approach, the multidimensional response is modeled as a GP vector using the same covariance function for each dimension. It accounts for correlations by introducing a constant correlation matrix between the outputs. However, in very high-dimensional settings (typical UQ applications have a few thousand outputs), dealing with the full correlation matrix is computationally challenging. Since in this work we are trying to develop a method that will be able to deal with output dimensions that range from a few hundreds to a few thousands, keeping the training time to acceptable levels is one of our major goals. We achieve this by making a compromise: the outputs will be treated as *conditionally independent* given the covariance function. Our approach is similar to that in [19] if a diagonal correlation matrix and a constant mean is used. The underlying assumption is that the regularity of all output dimensions is approximately the same. Since each output may vary in signal strength (e.g. finite element nodes close to a fixed boundary condition exhibit smaller variations compared to ones in the middle of the domain), we have to work with a scaled version of the responses. The computational savings of using a single covariance function for all outputs are tremendous: only a single Cholesky decomposition is required, dropping the training cost back to $O(N^3)$. We call the resulting model a Multi-output Gaussian Process (MGP) and refer to regression using MGPs as MGPR.

Let us introduce the *observed mean*:

$$\mu_{\text{obs},r} = \frac{1}{N} \sum_{n=1}^{N} y_r^{(n)}, \tag{2.8}$$

and the *observed variance*:

$$\sigma^2_{\text{obs},r} = \frac{1}{N}\sum_{n=1}^{N}(y_r - \mu_{\text{obs},r})^2, \tag{2.9}$$

of the data $\mathcal{D}$. We will be modeling the *scaled response functions* $g_r : \mathbf{X}^i \rightarrow \mathbb{R}$, defined by:

$$g_r(\mathbf{x}) = \frac{f_r(\mathbf{x}) - \mu_{\text{obs},r}}{\sigma_{\text{obs},r}}, r = 1, \dots, M. \tag{2.10}$$

The scaling is necessary, because the various outputs might exhibit different signal strengths. Obviously, this definition depends on the actual observations. We expect, however, that if N is big or if the stochastic element under investigation is small, then it is a good approximation to the ideal scaling, i.e. zero mean and unit variance for all outputs. Assuming that all outputs have the same regularity, we model each g_r as a Gaussian Process with zero mean and covariance function $c(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta})$:

$$g_r(\mathbf{x})|\boldsymbol{\theta} \sim \mathcal{GP}\left(0, c(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta})\right), r = 1, \dots, M,$$

where $\boldsymbol{\theta} \in \boldsymbol{\Theta} \subset \mathbb{R}^S$ are the $S \geq 1$, unknown *hyper-parameters* of the covariance function. That is, the scaled responses are treated as conditionally independent given the hyper-parameters.

Point Estimates of the Hyper-parameters A fully Bayesian approach would proceed by imposing a prior probability $\pi(\boldsymbol{\theta})$ over the hyper-parameters and then average (numerically) over them. Instead, we will employ the *evidence approximation* to Bayesian inference [**?**], in order to obtain point-estimates of the hyper-parameters by maximizing the marginal likelihood of the data (Ch. 5 of [77]). This necessarily underestimates the prediction uncertainty, but it is a trade-off we are willing to make in order to obtain a computationally tractable model. The logarithm of the marginal likelihood of each scaled response $g_r(\cdot), r = 1, \dots, M$ is given by:

$$\log p(\mathbf{z}_r|\mathcal{D}, \boldsymbol{\theta}) = -\frac{1}{2}\mathbf{z}_r^T \mathbf{C}^{-1}\mathbf{z}_r - \frac{1}{2}\log|\mathbf{C}| - \frac{N}{2}\log 2\pi,$$

where $\mathbf{z}_r = (z_r^{(1)}, \dots, z_r^{(N)})$ is a scaled version of the observations in $\mathcal{D}$:

$$z_r^{(n)} = \frac{y_r^{(n)} - \mu_{\text{obs},r}}{\sigma_{\text{obs},r}}, n = 1, \dots, N, \tag{2.11}$$

$\mathbf{C} = (C_{ij}), C_{ij} = c(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}; \boldsymbol{\theta})$ is the covariance matrix and $|\mathbf{C}|$ its determinant. Since the scaled responses are conditionally independent given $\boldsymbol{\theta}$, the logarithm of the joint marginal likelihood is simply the sum of the marginal likelihoods of each output, i.e.

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\theta}) &:= \log p(\mathbf{z}_1, \dots, \mathbf{z}_M|\mathbf{X}, \boldsymbol{\theta}) && (2.12)\\
&= \sum_{r=1}^{M}\log p(\mathbf{z}_r|\mathbf{X}, \boldsymbol{\theta}) && (2.13)\\
&= -\frac{1}{2}\sum_{r=1}^{M}\mathbf{z}_r^T\mathbf{C}^{-1}\mathbf{z}_r - \frac{M}{2}\log|\mathbf{C}| - \frac{NM}{2}\log 2\pi. && (2.14)
\end{aligned}$$

Thus, a point estimate of $\boldsymbol{\theta}$ over the element $\mathbf{X}^i$ is obtained by

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\theta}). \tag{2.15}$$

The joint marginal likelihood $\mathcal{L}(\boldsymbol{\theta})$ might exhibit multiple maxima which correspond to alternative interpretations of the data. In practice, we make an educated initial guess and we are satisfied with the (local) maximum obtained using a Conjugate Gradient method [76]. The specifics of the optimization method are discussed in Appendix A.

The Predictive Distribution Having decided on a point estimate for the hyper-parameters $\boldsymbol{\theta}$, we are ready to predict the scaled response at any test point

$\mathbf{x} \in \mathbf{X}^i$. Scaling back to the original responses, we can easily see that the predictive distribution of $f_r(\mathbf{x})$ is:

$$f_r(\mathbf{x})|\mathcal{D}, \boldsymbol{\theta}^* \sim \mathcal{N}\left(\mu_{f_r}(\mathbf{x}; \boldsymbol{\theta}^*), \sigma^2_{f_r}(\mathbf{x}; \boldsymbol{\theta}^*)\right), \tag{2.16}$$

with mean:

$$\mu_{f_r}(\mathbf{x}; \boldsymbol{\theta}^*) = \sigma_{\text{obs},r}\mathbf{c}^T\mathbf{C}^{-1}\mathbf{z}_r + \mu_{\text{obs},r}, \tag{2.17}$$

and variance:

$$\sigma^2_{f_r}(\mathbf{x}; \boldsymbol{\theta}^*) = \sigma^2_{\text{obs},r}\left(c(\mathbf{x}, \mathbf{x}; \boldsymbol{\theta}^*) - \mathbf{c}^T\mathbf{C}^{-1}\mathbf{c}\right), \tag{2.18}$$

where $\mathbf{c} = (c(\mathbf{x}, \mathbf{x}^{(1)}; \boldsymbol{\theta}^*), \dots, c(\mathbf{x}, \mathbf{x}^{(N)}; \boldsymbol{\theta}^*))$ and the covariance matrix $\mathbf{C}$ is evaluated at $\boldsymbol{\theta}^*$. We will refer to $\sigma^2_{f_r}(\mathbf{x}; \boldsymbol{\theta}^*)$ as the *predictive variance* of the response at $\mathbf{x}$. It represents our uncertainty about the prediction at this particular test point.

As mentioned earlier, the *predictive mean* $\mu_{f_r}(\mathbf{x}; \boldsymbol{\theta}^*)$ given by Equation 2.17 will be used to provide estimates for the statistics over the element $\mathbf{X}^i$, while the predictive variance $\sigma^2_{f_r}(\mathbf{x}; \boldsymbol{\theta}^*)$ will give error bars (see Section 2.2.2). Notice that $\mu_{f_r}(\mathbf{x}; \boldsymbol{\theta}^*)$ is, in fact, a *kernel estimator* since:

$$\mu_{f_r}(\mathbf{x}; \boldsymbol{\theta}^*) = \sum_{n=1}^{N} \alpha_{rn} c(\mathbf{x}^{(n)}, \mathbf{x}; \boldsymbol{\theta}^*) + \mu_{\text{obs},r}, \tag{2.19}$$

where the weights α_{rn} are given by:

$$\boldsymbol{\alpha}_r \equiv (\alpha_{r1}, \alpha_{r2}, \dots, \alpha_{rN}) := \sigma_{\text{obs},r}\mathbf{C}^{-1}\mathbf{z}_r,$$

and also depend on $\boldsymbol{\theta}^*$ through $\mathbf{C}$.

2.2.2 Calculation of the local statistics

As in the previous section, we focus on a specific element $\mathbf{X}^i$. All quantities are again local to $\mathbf{X}^i$. In order to keep notational complexity to a minimum, we do not explicitly show this dependence. We will derive *analytic* point estimates as well as error bars for the mean and the higher moments of the response based on the linear point estimator of $\mathbf{f}(\cdot)$ over $\mathbf{X}^i$ given in Equation 3.21 and the predictive variance Equation 3.22. To be exact, we are interested in estimating all moments $\mathbf{m}^q = (m_1^q, \dots, m_M^q), q \geq 1$, where

$$m_r^q = \int_{\mathbf{X}^i} f_r^q(\mathbf{x}) p^i(\mathbf{x}) d\mathbf{x}. \tag{2.20}$$

$p^i : \mathbf{X} \to \mathbb{R}$ is the *conditional probability density* related to $\mathbf{X}^i$:

$$p^i(\mathbf{x}) := \frac{p(\mathbf{x})}{P(\mathbf{X}^i)} 1_{\mathbf{X}^i}(\mathbf{x}), \tag{2.21}$$

where $P(\mathbf{X}^i)$ is the probability of an input point residing in the stochastic element $\mathbf{X}^i$, i.e.

$$P(\mathbf{X}^i) = \int_{\mathbf{X}^i} p(\mathbf{x}) d\mathbf{x}.$$

In order to achieve analytic estimates of $\mathbf{m}^q$, we keep concurrent MGP estimates of the response raised to the q power. In particular, the q power of the response is treated also as a MGP with its own hyper-parameters $\boldsymbol{\theta}^q$. Let us denote the predictive distribution for the q power of the response at $\mathbf{x} \in \mathbf{X}^i$ by:

$$f_r^q(\mathbf{x}) | \mathcal{D}, \boldsymbol{\theta}^q \sim \mathcal{N}\left(\mu_{f_r^q}(\mathbf{x}; \boldsymbol{\theta}^q), \sigma^2_{f_r^q}(\mathbf{x}; \boldsymbol{\theta}^q)\right),$$

where $\mu_{f_r^q}(\mathbf{x}; \boldsymbol{\theta}^q)$ is the predictive mean and $\sigma^2_{f_r^q}(\mathbf{x}; \boldsymbol{\theta}^q)$ the predictive variance for $r = 1, \dots, M$. These quantities are available through the exact same procedure described in Section 2.2.1, using the q power of the response instead of the response itself. For convenience, let us write the predictive mean at $\mathbf{x}$ as:

$$\mu_{f_r^q}(\mathbf{x}; \boldsymbol{\theta}^q) = \sum_{n=1}^N \alpha_{rn}^q c(\mathbf{x}^{(n)}, \mathbf{x}; \boldsymbol{\theta}^q) + \mu^q_{\mathrm{obs},r},$$

and the predictive variance at $\mathbf{x}$ as:

$$\sigma^2_{f_r^q}(\mathbf{x}; \boldsymbol{\theta}^q) = (\sigma^q_{\mathrm{obs},r})^2 \left(c(\mathbf{x}, \mathbf{x}; \boldsymbol{\theta}^q) - \mathbf{c}^{q,T} (\mathbf{C}^q)^{-1} \mathbf{c}^q\right),$$

where $\mu^q_{\text{obs},r}$ and $\sigma^q_{\text{obs},r}$ are defined as in Eqs. (3.9) and (3.10), respectively, using the q power of the observed response, $\mathbf{c}^q = (c(\mathbf{x}^{(1)}, \mathbf{x}; \boldsymbol{\theta}^q), \dots, c(\mathbf{x}^{(N)}, \mathbf{x}; \boldsymbol{\theta}^q))$ and $\mathbf{C}^q$ is the covariance matrix evaluated at $\boldsymbol{\theta}^q$.

Our goal is to derive a predictive probability distribution for the moments $\mathbf{m}^q$ given the data and the hyper-parameters. In a proper probabilistic treatment, we would proceed by sampling the full posterior of the MGP, integrating the samples over $\mathbf{x}$ and producing a Monte Carlo estimate of the predictive mean and variance of each moment. To obtain analytic estimates, let us make the simplifying assumption that the predictions at different input points $\mathbf{x}$ are conditionally independent given the data and the hyper-parameters. Then, by the additivity of independent normal variables, we arrive at the approximation:

$$m^q_r | \mathcal{D}, \boldsymbol{\theta}^q \sim \mathcal{N}\left(\mu_{m^q_r}, \sigma^2_{m^q_r}\right), \tag{2.22}$$

where the predictive mean of m^q_r is:

$$\mu_{m^q_r} = \int_{\mathbf{X}^i} \mu_{f^q_r}(\mathbf{x}; \boldsymbol{\theta}^q) p^i(\mathbf{x}) d\mathbf{x}, \tag{2.23}$$

and its predictive variance:

$$\sigma^2_{m^q_r} = \int_{\mathbf{X}^i} \sigma^2_{f^q_r}(\mathbf{x}; \boldsymbol{\theta}^q) p^i(\mathbf{x}) d\mathbf{x}. \tag{2.24}$$

Fortunately, the integrals involved can be expressed in terms of expectations of the covariance function with respect to the conditional input distribution. This results in a fast, semi-analytic estimate of $\mu_{m^q_r}$ and $\sigma_{m^q_r}$. It is worth mentioning at this point that this distribution is necessarily wider than the optimum one.

Remark 1 Obviously, the assumption that a positive function, e.g. the response f_r raised to an even power, is a Gaussian Process is not optimal, since the predictive distribution assigns positive probability to the event of the function getting negative values. However, this assumption is necessary in order to obtain analytic estimates of the predictive distribution of the statistics. A direct consequence of it is that the predictive distribution Equation 2.22 for an even moment has also positive probability of being negative. A tighter predictive distribution can always be found by truncating Equation 2.24 below zero. On the other hand, the predictive mean of an even moment will always be positive.

Evaluation of the integrals We now proceed to the calculation of the integrals in Eqs. (3.28) and (2.24). We can write the following:

$$\mu_{m_r^q} = \sum_{n=1}^{N} \alpha_{rn}^q \epsilon_n^q + \mu_r^q, \tag{2.25}$$

and

$$\sigma^2_{m_r^q} = (\sigma^q_{\text{obs},r})^2 \left(c^q - \sum_{n,l=1}^{N} (\mathbf{C}^q)^{-1}_{nl} \nu^q_{nl} \right), \tag{2.26}$$

where

$$\epsilon_n^q = \int_{\mathbf{X}^i} c(\mathbf{x}^{(n)}, \mathbf{x}; \boldsymbol{\theta}^q) p^i(\mathbf{x}) d\mathbf{x}, \tag{2.27}$$

$$c^q = \int_{\mathbf{X}^i} c(\mathbf{x}, \mathbf{x}; \boldsymbol{\theta}^q) p^i(\mathbf{x}) d\mathbf{x}, \tag{2.28}$$

$$\nu^q_{nm} = \int_{\mathbf{X}^i} c(\mathbf{x}, \mathbf{x}^{(n)}; \boldsymbol{\theta}^q) c(\mathbf{x}, \mathbf{x}^{(l)}; \boldsymbol{\theta}^q) p^i(\mathbf{x}) d\mathbf{x}, \tag{2.29}$$

and $(\mathbf{C}^q)^{-1}_{nl}$ is the nl element of the inverse q covariance matrix $(\mathbf{C}^q)^{-1}$.

Thus, computation of the statistics requires the evaluation of integrals of the form of Eqs. (2.27), (2.28) and (2.29). In Appendix A, we provide analytic formulas for their calculation for the special case of uniform input distribution and Squared Exponential (SE) covariance function. For the SE covariance function but arbitrary input probability density of the form of Equation 2.1, their evaluation requires $O(K)$ one-dimensional numerical integrations.

2.2.3 From local to global statistics

In the same spirit as the multi-element methods [91, 92, 28], we combine the statistics over each stochastic element in order to obtain their global analogues. Since we now work over the whole domain, we will explicitly mark the dependence of the underlying quantities on the element $\mathbf{X}^i, i = 1, \dots I$. Let $m_r^{q,i}$ be the q moment of the response that pertains to the conditional probability density $p^i(\mathbf{x})$ (Equation 3.27) and m_r^q be the global one (Equation 3.1). Notice that m_r^q can be decomposed as

$$
\begin{aligned}
m_r^q &= \int_{\mathbf{X}} f_r^q(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} &(2.30)\\
&= \sum_{i=1}^{I} \int_{\mathbf{X}^i} f_r^q(\mathbf{x}) \frac{p(\mathbf{x})}{P(\mathbf{X}^i)} d\mathbf{x} P(\mathbf{X}^i) &(2.31)\\
&= \sum_{i=1}^{I} \int_{\mathbf{X}^i} f_r^q(\mathbf{x}) p^i(\mathbf{x}) d\mathbf{x} P(\mathbf{X}^i), &(2.32)
\end{aligned}
$$

or

$$
m_r^q = \sum_{i=1}^{I} m_r^{q,i} P(\mathbf{X}^i). \tag{2.33}
$$

Now, assume that for each element $\mathbf{X}^i, i = 1, \dots I$ we have obtained a predictive distribution (Equation 2.22) for $m_r^{q,i}$ and let its predictive mean and variance be $\mu_{m_r^{q,i}}$ and $(\sigma_{m_r^{q,i}})^2$, respectively (Eqs. (2.25) and (2.26)). Assuming conditional independence of the predictive distributions given the data and the hyper-parameters, we obtain that:

$$
m_r^q | \mathcal{D}, \boldsymbol{\theta}^q \sim \mathcal{N}\left(\mu_{m_r^q}, \sigma^2_{m_r^q}\right), \tag{2.34}
$$

where the predictive mean is:

$$
\mu_{m_r^q} = \sum_{i=1}^{I} \mu_{m_r^{q,i}} P(\mathbf{X}^i), \tag{2.35}
$$

and the predictive variance:

$$\sigma^2_{m_r^q} = \sum_{i=1}^{I} \sigma^2_{m_r^{q,i}} P(\mathbf{X}^i). \tag{2.36}$$

Again, truncation of this distribution below zero for even q, always yields an improved estimator (see Remark 1).

Finally, we derive a normal approximation to the predictive distribution for the variance of the response $\mathbf{v} = (v_1, \ldots, v_M)$ (defined in Equation 3.3):

$$v_r \sim \mathcal{N}\left(\mu_{v_r}, \sigma^2_{v_r}\right). \tag{2.37}$$

Under the assumption of conditional independence of $m_r^q,\ q = 1, 2$, the predictive mean of v_r is given by:

$$\begin{aligned} \mu_{v_r} &:= \mathbf{E}\left[m_r^2 - (m_r^1)^2 | \mathcal{D}, \boldsymbol{\theta}^1, \boldsymbol{\theta}^2\right] && (2.38) \\ &= \mathbf{E}\left[m_r^2 \middle| \mathcal{D}, \boldsymbol{\theta}^1, \boldsymbol{\theta}^2\right] - \mathbf{E}\left[(m_r^1)^2 | \mathcal{D}, \boldsymbol{\theta}^1, \boldsymbol{\theta}^2\right], && (2.39) \end{aligned}$$

or:

$$\mu_{v_r} = \mu_{m_r^2} - \mu^2_{m_r^1} - \sigma^2_{m_r^1}, \tag{2.40}$$

where $\mathbf{E}[\cdot|\mathcal{D}, \boldsymbol{\theta}^1, \boldsymbol{\theta}^2]$ denotes the expectation with respect to the joint predictive distribution for m_r^1 and m_r^2. Equivalently, the predictive variance is:

$$\begin{aligned} \sigma^2_{v_r} &:= \mathbf{V}\left[m_r^2 - (m_r^1)^2 | \mathcal{D}, \boldsymbol{\theta}^1, \boldsymbol{\theta}^2\right] && (2.41) \\ &= \mathbf{V}\left[m_r^2 | \mathcal{D}, \boldsymbol{\theta}^1, \boldsymbol{\theta}^2\right] + \mathbf{V}\left[(m_r^1)^2 | \mathcal{D}, \boldsymbol{\theta}^1, \boldsymbol{\theta}^2\right] && (2.42) \\ &= \mathbf{V}\left[m_r^2 | \mathcal{D}, \boldsymbol{\theta}^1, \boldsymbol{\theta}^2\right] + \mathbf{E}\left[(m_r^1)^4 | \mathcal{D}, \boldsymbol{\theta}^1, \boldsymbol{\theta}^2\right] - \left(\mathbf{E}\left[(m_r^1)^2 | \mathcal{D}, \boldsymbol{\theta}^1, \boldsymbol{\theta}^2\right]\right)^2, && (2.43) \end{aligned}$$

or:

$$\sigma^2_{v_r} = \sigma^2_{m_r^2} + 4\mu^2_{m_r^1}\sigma^2_{\mu_r^1} + 2\sigma^4_{\mu_r^1}, \tag{2.44}$$

where $\mathbf{V}[\cdot|\mathcal{D}, \boldsymbol{\theta}^1, \boldsymbol{\theta}^2]$ denotes the variance with respect to the joint predictive distribution of m_r^1 and m_r^2.

Let us end this section by mentioning that the above procedure can be easily applied to obtain normal approximations to the predictive distributions of any centered moment. It is obvious that the calculation can always be casted in terms of moments of the normal distribution which are readily available using the *confluent hypergeometric function* $U(a, b, x)$ (see Ch. 13 of [2]).

2.2.4 Adaptivity

In this section, we develop an iterative procedure to adaptively decompose the stochastic space in smaller elements. The initial step of this procedure starts by considering a single element, i.e. $\mathbf{X}$ itself. Here, we assume that we are already given a decomposition of the domain as well as a local surrogate model on each element. The decision we wish to make is whether or not to refine a given element and in which way. We develop refinement criteria that are based solely on information gathered by the current surrogate model and no further calls to the deterministic solver are required. The Bayesian predictive variance Equation 3.22 is used to define a measure of our uncertainty about the prediction over the whole domain $\mathbf{X}$. We show how this measure can be broken down to contributions coming from each element. Based on this observation, we derive a criterion that suggests refinement of an element if its contribution to the global uncertainty is larger than a pre-specified threshold. For the sake of simplicity, we only consider rectangular elements and refine them by splitting them perpendicular to the dimension of greatest importance in two pieces of equal probability. The importance of a particular dimension is characterized by its length scale. The length scales are identified as the hyper-parameters of a SE covariance function.

Suppose that we have already a decomposition of the stochastic domain $\mathbf{X}$ in *rectangular* elements $\mathbf{X}^i$, e.g.

$$\mathbf{X}^i = [a_1^i, b_1^i] \times \cdots \times [a_K^i, b_K^i],$$

with $a_k^i < b_k^i, k = 1, \ldots, K, i = 1, \ldots, I$ such that Equation 3.4 holds. Furthermore, assume that we have already learnt the local surrogates on each element $\mathbf{X}^i$. Let $\sigma^2_{f_r^i}(\mathbf{x})$ be the predictive variance of the $r = 1, \ldots, M$ output of the local surrogate of $\mathbf{f}^i$ at $\mathbf{x} \in \mathbf{X}^i$ (Equation 3.22). By the conditional independence assumption for the predictive distribution over each element and Equation 3.5, the predictive variance of the $r = 1, \ldots, M$ dimension of the global surrogate $\sigma^2_{f_r}(\mathbf{x})$ at $\mathbf{x} \in \mathbf{X}$ is given by:

$$\sigma^2_{f_r}(\mathbf{x}) = \sum_{i=1}^{I} \sigma^2_{f_r^i}(\mathbf{x}) 1_{\mathbf{X}^i}(\mathbf{x}). \tag{2.45}$$

Its average over r,

$$\sigma^2_{\mathbf{f}}(\mathbf{x}) := \frac{1}{M} \sigma^2_{f_r^i}(\mathbf{x}),$$

is a measure of our uncertainty about the prediction of all outputs simultaneously at the test point $\mathbf{x} \in \mathbf{X}$. Taking the expectation of this quantity with respect to the input probability density $p(\mathbf{x})$, we obtain

$$\sigma^2_{\mathbf{f},p} := \int_{\mathbf{X}} \sigma^2_{\mathbf{f}}(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}. \tag{2.46}$$

This quantity is a measure of our uncertainty about our prediction over the whole domain $\mathbf{X}$. Notice that, in $\sigma^2_{\mathbf{f},p}$, the uncertainty of the model at $\mathbf{x}$ is weighted by its probability of occurrence $p(\mathbf{x})$. Intuitively speaking, we are willing to accept a somewhat less accurate surrogate in regions of the space occurring with lower probability. Using Equation 3.35, it is straightforward to see that:

$$\sigma^2_{\mathbf{f},p} = \sum_{i=1}^{I} \sigma^2_{\mathbf{f},p^i} P(\mathbf{X}^i), \tag{2.47}$$

where

$$\sigma^2_{\mathbf{f},p^i} := \int_{\mathbf{X}^i} \sigma^2_{\mathbf{f}}(\mathbf{x}) p^i(\mathbf{x}) d\mathbf{x},$$

is the uncertainty of our prediction over the element $\mathbf{X}^i$. Making use of Equation 2.24 for $q = 1$, we obtain that:

$$\sigma^2_{\mathbf{f},p^i} = \frac{1}{M} \sum_{r=1}^{M} \sigma^2_{m_r^{1,i}}. \tag{2.48}$$

Hence, $\sigma^2_{\mathbf{f},p^i}$ relates directly to our uncertainty about the mean response $\sigma^2_{m_r^{1,i}}$ (Equation 2.26). Generalizing, we can define the corresponding uncertainties for the response raised to the $q \geq 1$ power (see Section 2.2.3):

$$\sigma^2_{\mathbf{f}^q,p} := \sum_{i=1}^{I} \sigma^2_{\mathbf{f}^q,p^i} P(\mathbf{X}^i), \tag{2.49}$$

where

$$\sigma^2_{\mathbf{f}^q,p^i} := \frac{1}{M} \sum_{r=1}^{M} \sigma^2_{m_r^{q,i}}. \tag{2.50}$$

This measure is equivalent to our uncertainty about the q-th moment of the response. Our idea it to refine the element $\mathbf{X}^i$, if the contribution to the global uncertainty coming from it, is greater than a certain threshold $\delta > 0$, that is we refine $\mathbf{X}^i$ if:

$$\sigma^2_{\mathbf{f}^q,p^i} P(\mathbf{X}^i) > \delta, \text{ for any } q = 1, 2, \dots, \tag{2.51}$$

depending on how many moments one wishes to consider. However, in the numerical examples of the present work, we simply use the criterion for $q = 1$, despite the fact that we report also the variance. We plan to investigate its dependence on q in a later work.

The above criterion specifies whether or not an element $\mathbf{X}^i$ should be refined. As already mentioned, we refine elements by cutting them in equally probable parts perpendicular to 'the most important dimension'. At this point, we attempt to give a precise meaning to the concept of 'the most important dimension'. Towards this goal, we will exploit the properties of a specific parametric form for the covariance function, the Squared Exponential (SE):

$$c_{\mathrm{SE}}(\mathbf{x}, \mathbf{x}') = s_f^2 \exp\left(-\frac{1}{2}\sum_{k=1}^{K}\frac{(x_k - x_k')^2}{\ell_k^2}\right), \tag{2.52}$$

where $s_f > 0$ can be interpreted as the *signal strength* and $\ell_k > 0$ as the *length scale* of each stochastic input. These parameters can be learnt from the data by using the evidence approximation (see Section 2.2.1), allowing the determination of the relative importance of each dimension. The technique is called *automatic relevance determination* (ARD). It originated in the Neural Networks literature [67] and was later extended to GP Regression [94]. We emphasize that a unique set of the SE hyper-parameters is learnt on each element $\mathbf{X}^i$ (as well as for each power of the response, $\mathbf{f}^q$, that we take into account). Hence, despite the fact that each local surrogate is a stationary GP, the global surrogate is non-stationary. This is similar in spirit to the Bayesian Treed Gaussian Process Model in [41].

Let us explicitly denote the learnt length scales of element $\mathbf{X}^i$ corresponding to the MGP that represents $\mathbf{f}$, with $\ell_k^i, k = 1, \dots, K$. The length scales of the powers of the response, $\mathbf{f}^q, q > 1$, are not involved in the criterion we are about to formulate. Furthermore, let us introduce the probability P_k^i that the k-th dimension x_k of a random input point $\mathbf{x} \in \mathbf{X}$ falls inside $\mathbf{X}^i$:

$$P_k^i := \int_{a_k^i}^{b_k^i} p_k(x_k)dx_k. \tag{2.53}$$

In general, this has to be evaluated numerically. For the special case of uniform distribution on a rectangular $\mathbf{X}$, we obtain:

$$P_k^i = \frac{b_k^i - a_k^i}{b_k - a_k}.$$

We define the *importance* I_k^i of the dimension k of the element $\mathbf{X}^i$ to be:

$$I_k^i = P_k^i / \ell_k^i. \tag{2.54}$$

Intuitively, the importance of a particular dimension is inversely proportional to the inferred length scale and proportional to the probability mass along that dimension trapped within the stochastic element. Thus, if $\mathbf{X}^i$ needs refinement (i.e. satisfies Equation 3.38), we cut it perpendicular to the most important dimension k^*, given by:

$$k^* = \arg\max_k I_k^i. \tag{2.55}$$

In order to have two new elements with the same probabilities of occurrence, the splitting point is given by the median of the marginal conditional distribution of $\mathbf{X}^i$ along dimension k, $p_k^i(x_k)$ defined by:

$$p_k^i(x_k) = \frac{p_k(x_k)}{\int_{a_k^i}^{b_k^i} p_k(x_k')dx_k'} 1_{[a_k^i, b_k^i]}(x_k). \tag{2.56}$$

This is a root finding problem that can easily be solved using a bisection algorithm. For the special case of the uniform distribution, the splitting point trivially becomes:

$$x_k^* = \frac{1}{2}(a_k^i + b_k^i).$$

Remark 2 The particular splitting criterion based on the inferred length scales is not the only possibility. Despite being intuitively appealing, it remains an ad hoc choice. Nevertheless, its computational evaluation time is negligible and we have empirically shown that it results in decompositions that concentrate around important features of the response. Of course, its performance depends crucially on predicting correctly the length scales.

2.2.5 Collection of the observations

In this section, we discuss how the data within an element are collected. We have to consider two distinct cases:

1. No data have been observed yet and we only have a single element (i.e. $\mathbf{X}$ itself).
2. We have obtained a fit of the response over an element $\mathbf{X}^i$ based on N^i observations
$$\mathcal{D}^i = \{(\mathbf{x}^{i,(n)}, \mathbf{y}^{i,(n)})\}_{n=1}^{N^i},$$
and we have decided to split it in two elements $\mathbf{X}^{i,1}$ and $\mathbf{X}^{i,2}$ so that
$$\mathbf{X}^i = \mathbf{X}^{i,1} \cup \mathbf{X}^{i,2} \text{ and } \mathbf{X}^{i,1} \cap \mathbf{X}^{i,2} = \emptyset.$$

Let $N \geq 1$ be the *maximum number of observations per element* we wish to consider within each element and $\delta > 0$ be the desired uncertainty tolerance of each element (see Equation 3.38). We deal with the first case (no observations made so far), by simply observing N random data points drawn from the input probability distribution $p(\mathbf{x})$. In the second case, we wish to utilize the MGP we already have for $\mathbf{X}^i$, in order to make the most informative selection of new data points. This procedure is known in the literature as *Experimental Design* (ED).

The ED problem can be formulated in a Bayesian framework in terms of maximizing the expectation of a utility function (see [14] for a good review of Bayesian ED). If we observe the data points one by one and update the model each time, then the procedure is termed *Sequential Experimental Design* (SED). In the machine learning literature SED is known as *Active Learning* (AL). According to MacKay [**?**], if the utility function we choose is the change in entropy of the posterior of the hyper-parameters $\boldsymbol{\theta}$, then - under the evidence approximation - the most informative input point corresponds to the one that maximizes the predictive variance of the model. This criterion is termed *Active Learning MacKay* (ALM). An alternative to ALM is Cohn's criterion (ALC) [18], which proceeds by choosing the input point that maximizes the expected change in output predictive variance over the whole domain. ALC has the advantage that it allows one to weight the input space by a probability distribution, which in our setting would naturally be the input probability distribution of the element $\mathbf{X}^i$. ALC has also been numerically shown to perform better than ALM (for a comparison of ALM and ALC see [83] and the corresponding discussion in [42]). However, ALC is not based on a decision theoretic foundation and it is much harder to implement. In this work - mainly for computational purposes - we choose to work with ALM. We now, describe its extension to the multi-output case.

We start, by splitting the observed data in two sets $\mathcal{D}^{i,l},\ l = 1, 2$ according to which element the inputs belong to, i.e.

$$\mathcal{D}^{i,l} = \{(\mathbf{x}, \mathbf{y}) \in \mathcal{D}^i : \mathbf{x} \in \mathbf{X}^{i,l}\},\ l = 1, 2.$$

Let $\boldsymbol{\theta}^*$ be the hyper-parameters of the MGP over $\mathbf{X}^i$ and $\sigma^2_{f_r}(\mathbf{x}; \boldsymbol{\theta}^*)$ be the corresponding predictive variance of the r-th output at $\mathbf{x} \in \mathbf{X}^i$ given by Equation 3.22. Throughout the SED procedure, the hyper-parameters will be kept constant. Without loss of generality, we work with the left child of $\mathbf{X}^i$, $\mathbf{X}^{i,1}$. The right child is treated similarly. We will be sequentially observing $\mathbf{x}^{\text{new},m}$ and the corresponding responses $\mathbf{y}^{\text{new},m} = \mathbf{f}(\mathbf{x}^{\text{new},m})$ for $m = 1, 2, \dots$. Let the set of observations residing in $\mathbf{X}^{i,1}$ be:

$$\mathcal{D}^{i,1,n} = \mathcal{D}^{i,1} \cup \{\mathbf{x}^{\text{new},m} : m = 1, \dots, n\}, n \geq 1,$$

where $\mathcal{D}^{i,1,0} = \mathcal{D}^{i,1}$. Denote by $\sigma^2_{f_r}(\mathbf{x};\boldsymbol{\theta}^*,\mathcal{D}^{i,1,n})$ the predictive variance of the r-th output when $\mathcal{D}^{i,1,n}$ is taken into account. From Equation 3.22, it is apparent that $\sigma^2_{f_r}(\mathbf{x};\boldsymbol{\theta}^*,\mathcal{D}^{i,1,n})$ depends only on the observed input points and not on the responses. Furthermore, since $\boldsymbol{\theta}^*$ remains constant, the inverse covariance matrix can be estimated sequentially at each step without the need to perform a Cholesky decomposition (see [61]). The extension of ALM to the multi-output case is as follows: given $\mathcal{D}^n_{i,1}$, observe the input point $\mathbf{x}^{\text{new},n+1} \in \mathbf{X}^{i,1}$ that maximizes the joint uncertainty of all outputs:

$$\sigma^2_{\mathbf{f}}(\mathbf{x};\boldsymbol{\theta}^*;\mathcal{D}^{i,1,n}) = \frac{1}{M}\sum_{r=1}^{M}\sigma^2_{f_r}(\mathbf{x};\boldsymbol{\theta}^*,\mathcal{D}^{i,1,n}). \tag{2.57}$$

That is,

$$\mathbf{x}^{\text{new},n+1} = \arg\max_{\mathbf{x}\in\mathbf{X}^{i,1}}\sigma^2_{\mathbf{f}}(\mathbf{x};\boldsymbol{\theta}^*;\mathcal{D}^{i,1,n}). \tag{2.58}$$

In an effort to introduce a bias from the input probability distribution, we suggest using:

$$\mathbf{x}^{\text{new},n+1} = \arg\max_{\mathbf{x}\in\mathbf{X}^{i,1}}\sigma^2_{\mathbf{f}}(\mathbf{x};\boldsymbol{\theta}^*;\mathcal{D}^{i,1,n})p(\mathbf{x}), \tag{2.59}$$

which causes low probability regions to be ignored. Of course, for the uniform case the two criteria are equivalent. We stop, either if N data points have been collected in $\mathcal{D}^{i,1,n}$, or if:

$$\sigma^2_{\mathbf{f},p^{i,1}}(\mathcal{D}^{i,1,n})P(\mathbf{X}^{i,1}) \le \delta, \tag{2.60}$$

where $\sigma^2_{\mathbf{f},p^{i,1}}(\mathcal{D}^{i,1,n})P(\mathbf{X}^{i,1})$ is the expectation of $\sigma^2_{\mathbf{f}}(\mathbf{x};\boldsymbol{\theta}^*;\mathcal{D}^{i,1,n})$ with respect to the conditional probability $p^{i,1}(\mathbf{x})$ of $\mathbf{X}^{i,1}$ (in the same spirit as it was used in Section 2.2.4).

The optimization problem in Equation 2.59 is relatively hard and involves several local maxima. Instead of solving it with a direct method, we use a simple Monte Carlo procedure to obtain an approximate solution. We draw $N_{textALM}$ random samples in $\mathbf{X}^{i,1}$, evaluate the product of the predictive variances and the input probability density (Equation 2.57) and select the one yielding the greatest result. This is affordable, since $\sigma^2_{\mathbf{f}}(\mathbf{x};\boldsymbol{\theta}^*;\mathcal{D}^{i,1,n})$ is cheap to evaluate.

2.2.6 A complete view at the framework

In this final section, we put together the building blocks of our scheme and discuss the algorithmic details and possible parallelization strategies. The basic input required is the maximum number of observations per element N and the tolerance $\delta > 0$, used for the refinement criterion (Equation 3.38) as well as the stopping criterion of ALM (Equation 2.60). An additional input is the number of MC samples used to approximate the solution to Equation 2.59 (last paragraph of Section 2.2.5), which we fix to $N_{\mathrm{ALM}} = 10000$.

Our scheme works in one element cycles that comprise of collecting observations (randomly or using ALM (Section 2.2.5)), fitting (Section 2.2.1) and adapting (Section 2.2.4). Let us denote with $\mathbf{X}^i$ a stochastic element, $\mathcal{D}^i$ the observations made on $\mathbf{X}^i$ and $\mathcal{M}^i$ the MGP fitted over $\mathbf{X}^i$ using $\mathcal{D}^i$. Let C be the set of triplets $(\mathbf{X}^i,\mathcal{D}^i,\mathcal{M}^i)$ for which the refinement criterion Equation 3.38 is not satisfied. We will refer to C as the set of *completed triplets*. The rest of the triplets are put in $\mathcal{U}$, called the set of *uncompleted triplets*. With $|\mathcal{D}^i|$ we denote the number of observations inside $\mathcal{D}^i$. Algorithm 3 provides a serial implementation of the scheme.

Parallelization of Algorithm 3 is relatively easy. Each node p, has its own set of completed C_p and uncompleted $\mathcal{U}_p$ elements. Initially the root node $p = 0$ starts as in Algorithm 3 and the rest with $\mathcal{U}_p = \emptyset, C_p = \emptyset, p \neq 0$. Then, everything proceeds as in Algorithm 3 with load re-balancing at the end of each outer

Algorithm 1: The complete surrogate building framework

$\mathcal{U} \leftarrow \{(\mathbf{X}, \emptyset, \emptyset)\}$.

$\mathcal{C} \leftarrow \emptyset$.

while $\mathcal{U} \neq \emptyset$ **do**

  Remove $(\mathbf{X}^i, \mathcal{D}^i, \mathcal{M}^i)$ from $\mathcal{U}$.

  if $\mathcal{M}^i = \emptyset$ **then**

    Observe N random points drawn from $p^i(\mathbf{x})$ Equation 2.21.

  else

    while $|\mathcal{D}^i| < N$ or Equation 2.60 not satisfied for δ **do**

      Add an observation to $\mathcal{D}^i$ using the ALM procedure (Equation 2.59).

      Update $\mathcal{M}^i$ to take into account the new data in $\mathcal{D}^i$.

    end while

  end if

  Refit the hyper-parameters of $\mathcal{M}^i$ using only the data in $\mathcal{D}^i$ (Section 2.2.1).

  if Refinement criterion of Equation 3.38 is satisfied for δ **then**

    Split $\mathbf{X}^i$ in $\mathbf{X}^{i,1}$ and $\mathbf{X}^{i,2}$ according to Equation 3.41.

    Let $\mathcal{D}^{i,1}$ and $\mathcal{D}^{i,2}$ to be the set observations residing in $\mathbf{X}^{i,1}$ and $\mathbf{X}^{i,2}$, respectively.

    $\mathcal{U} \leftarrow \mathcal{U} \cup \{(\mathbf{X}^{i,1}, \mathcal{D}^{i,1}, \mathcal{M}^i), (\mathbf{X}^{i,2}, \mathcal{D}^{i,2}, \mathcal{M}^i)\}$.

  else

    $\mathcal{C} \leftarrow \mathcal{C} \cup \{(\mathbf{X}^i, \mathcal{D}^i, \mathcal{M}^i)\}$.

  end if

end while

iteration (uncompleted elements are sent to processors with $\mathcal{U}_p = \emptyset$).

Remark 3 The choice of the maximum number of samples per element N is a crucial parameter to the scheme. Its optimal value depends in a complicated way on the (a priori unknown) smoothness of the underlying response as well as the number of hyper-parameters S. Its importance is more evident on the very first element of the scheme because it drives the rest of the tree construction as well as the Active Learning procedure. If a small value is used, then local features may be lost, while a very big value may result in redundant information. Similar problems are present in practically all UQ schemes. For example, ME-gPC depends on the polynomial degree and ASGC depends on which level of the Sparse Grid is adaptivity initiated. On the other hand, N makes our method computationally tractable, since it bounds above the dimensions of the covariance matrices that need to be inverted. A theoretical analysis of the optimal value of N is highly desirable, but clearly beyond the scope of the present work. In the engineering problems that we are interested in, one usually already has a rough idea about the smoothness of the problem based on some preliminary simulations. For smooth problems, using $N \approx 2K$, where K is the number of input dimensions gives satisfying results (see the Elliptic and Natural Convection numerical examples in Section **??**). For problems with local features, a slightly bigger value must be used. Empirically, we fix δ to a high value (e.g. $\delta \approx 10^{-1}$), we start with $N = 2K$ and increase N gradually until the results do not change any more. For this final N, we decrease δ and resume the scheme.

2.3 Numerical Examples

All examples are run on massively parallel computers at the National Energy Research Scientific Computing Center (NERSCC). The parallelization strategy is straightforward: each processor is assigned to work with a single element. The communication burden between the processes is minimal. Our implementation utilizes extensively the Trilinos library [48] as well as GSL [30].

The ultimate goal of the numerical examples is to demonstrate that the method can:

1. Learn non-stationary surfaces.
2. Deal with discontinuities.
3. Identify localized features of the response.
4. Reduce sampling frequency on unimportant input dimensions.

Whenever possible, we will compare our results with Sparse Grid Collocation (SGC) and Adaptive Sparse Grid Collocation (ASGC) [59]. Each method will be evaluated by considering an error measure of the predictive surface or of the statistics, as a function of the number of sample points used. In Sections 2.3.2, 2.3.3, 2.3.4, we apply our method to UQ problems. In all problems, the underlying input probability distribution $p(\mathbf{x})$ is understood to be the uniform distribution over the input domain, unless otherwise stated. The covariance function we use, is the SE with a nugget $g^2 = 10^{-6}$ (The nugget is required for numerical stability. See the discussion in Appendix A for more details.). All tasks start with a single element (the input domain itself) and N random samples drawn from the input distribution. N, is also the maximum number of samples taken within an element and is different for each example (See Remark 3 for how N is chosen). From that point, the algorithm proceeds until a pre-specified tolerance $\delta > 0$ is reached. The refinement criterion is given by Equation 3.38 for $q = 1$. The same tolerance δ is used to stop the ALM procedure of Section 2.2.5

(see Equation 2.60). The solution to the optimization problem of ALM (Equation 2.59) is approximated by drawing $N_{\mathrm{ALM}} = 10000$ samples in $\mathbf{X}^i$, evaluating $\sigma^2_{\mathbf{f}}(\mathbf{x}; \boldsymbol{\theta}^*; \mathcal{D}^{i,1,n})p_i(\mathbf{x})$ and selecting the one with the maximum value. The parameters of the method are g, N, N_{ALM} and δ.

2.3.1 Simple Validation Example

We start by applying our scheme to the problem of learning the single-output synthetic function (introduced in [42]):

$$f(x_1, x_2) = x_1 \exp\left\{-x_1^2 - x_2^2\right\}, \tag{2.61}$$

on $\mathbf{X} = [-2, 6]^2$. The input probability distribution is assumed to be uniform. This function is peculiar, in the sense that it has two localized features inside the box $[-2, 2]^2$, while it is practically zero everywhere else. The reason that we choose to begin our numerical examples with this toy problem are: 1) Its computational simplicity allows us to thoroughly test the dependence of our scheme on N (maximum number of samples per element) and 2) Its single output nature allows a direct comparison with the Treed Gaussian Process (TGP) of [41] which utilizes a Bayesian tree inspired by the Bayesian CART model.

The performance of each run is evaluated by comparing the predictive mean $\mu_f(\mathbf{x})$ to the true response. The error measure of choice here is the Mean Square Error (MSE) of $S = 10^5$ uniform random samples in $[-2, 6]^2$. Specifically, the MSE is defined to be

$$\mathrm{MSE}\left(\mu_f(\cdot)\right) := \frac{1}{S}\sum_{s=1}^{S}\left(\mu_f(\mathbf{x}^{(s)}) - f(\mathbf{x}^{(s)})\right)^2, \tag{2.62}$$

where $\mathbf{x}^{(s)}, s = 1, \dots, S$ are the random samples. Since, these samples were not used in the fitting procedure, the MSE is a measure of the predictive capabilities of the regression method.

Dependence of the results on the choice of N The choice of N in this example plays an important role since it determines the starting point of our algorithm. In order to test the sensitivity of the results on N, we run our scheme for $N = 10, 20, 30, 40, 50, 60, 70, 80$ and 90 until a tolerance of $\delta = 10^{-3}$ is reached. Figure 2.1 shows how the MSE depends on the number of observations made at the initial stage as well when the desired tolerance has been reached. We observe that for $N = 10, 20$ and 30, the scheme cannot represent the surface accurately. Because the features are very localized, such a small N is not enough to discover them. As a result the model for these choices of N is overconfident. The convergence rate is gradually improved reaching an optimum for $N = 70$ and then it deteriorates. However, the performance of the choices $N = 40, 50, 60, 70, 80$ and 90 as captured by the MSE as a function of the number of observations is relatively stable. Furthermore, the trees obtained are exactly the same (two of them are shown in Figs. 2.2 (a) and (c)). Finally, the third column of Table 2.1 shows the computational time (in seconds) taken by each one of these runs. It is apparent that smaller N reduces the computational cost of training the model. Let us conclude by stating that the choice of N is indeed a major decision in applying our scheme. Picking a very small value will improve the training time but increase the probability of not observing the important features. Picking a very large value will reveal more about the response surface but will slow down the training procedure. The optimum choice of N depends on the regularity of the response as well as on how localized its features are. In practice, however, this problem can be circumvented by initializing the scheme with more than one elements based on the number of available initial observations and the choice of N that one can afford (larger N results in larger training times). Of course, this requires observing a few realizations prior to running the scheme. Similar phenomena have been observed when one tries to apply ASGC to such a problem. In this case starting from a low level interpolation will make the algorithm believe that the surface is more regular than it actually is.

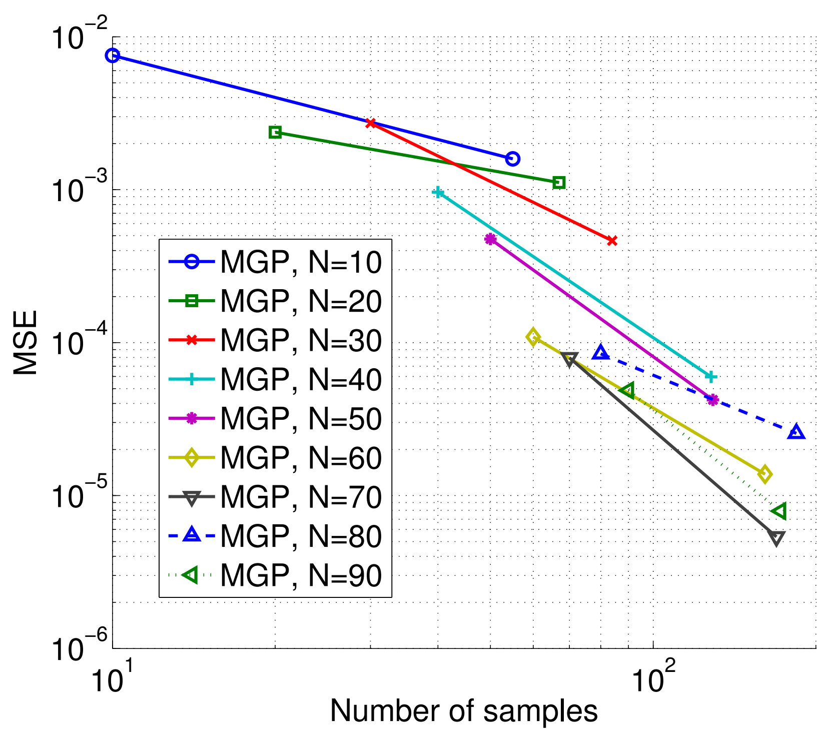


Figure 2.1: Validation example: The MSE in the prediction of $f(\mathbf{x})$ as a function of the number of observed samples for MGP for various choices of N. The MSE is calculated at the initial stage (one element) and at the final stage (when a tolerance of $\delta = 10^{-3}$ has been reached).

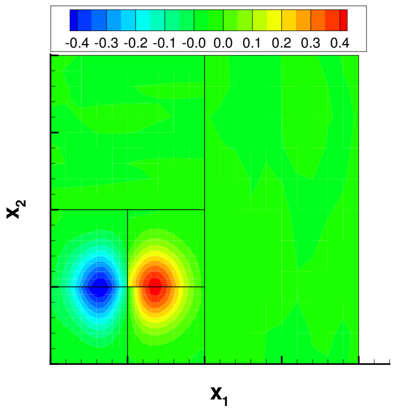
Comparison with Other Methodologies We start by comparing the performance of our scheme to SGC. Figure 2.3 plots the MSE for MGP and SGC as a function of the number of observations. We report our results only for $N = 50$.

Model	Observations	Time (sec)
MGP, $N = 20$	67	23
MGP, $N = 30$	84	45
MGP, $N = 40$	128	85
MGP, $N = 50$	129	102
MGP, $N = 60$	161	150
MGP, $N = 70$	169	168
MGP, $N = 80$	184	234
MGP, $N = 90$	172	186
TGP (ALM)	150	1472

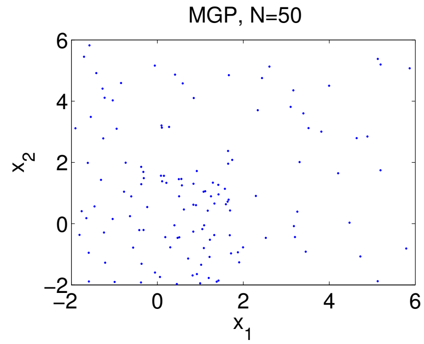
Table 2.1: Validation example: The table shows the number of observations required for the model to reach a tolerance of $\delta = 10^{-3}$ as well as the computational cost in seconds for the MGP model for N ranging from 20 to 90. We also report the computational time taken by the TGP model in order to sequentially gather 150 observations using the ALM criterion.

We note that SGC requires a very large number of observations. The MSE of ASGC is not reported since it fails to identify the localized features when it starts from Level 1. The observations shown for MGP correspond to tolerances of $\delta = 10^{-3}, 10^{-4}, 10^{-5}$ and 10^{-6}. We notice that SGC is out-performed by more than two orders of magnitude.

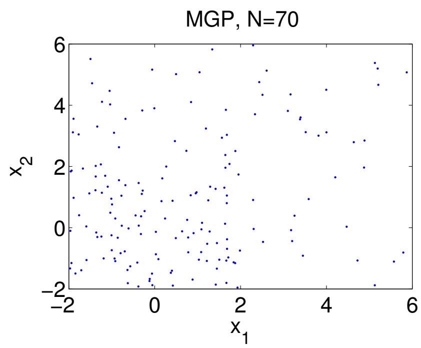
Finally, we compare the tree structures we obtain to the ones computed by TGP [41]. To the best of our knowledge, TGP is the only model of treed Gaussian Process using a fully Bayesian tree. We use the R implementation of the TGP model developed by Gramacy and Taddy which can be found in http://cran.r-project.org/web/packages/tgp/index.html. In order to make a fair comparison of TGP and our scheme, we used 1) the same computer for all tests, 2) the same LAPACK libraries and 3) a single CPU version of our code. We start by training the TGP model using 50 random samples and then sequentially add new samples using the ALM criterion (choosing from a pool of $10,000$ random candidates) until 150 observations have been gathered. As is shown in Figure 2.2 (e), TGP discovers three important regions which are very similar to the six regions discovered by MGP (see Figs. 2.2 (a) and (c)). In the same figure we also notice that the observations gathered by both TGP and MGP are quite similar. Table 2.1 compares the computational time of MGP for various N with that of TGP. Despite the fact that both models result in similar trees, it is observed that MGP is approximately 10 times faster than TGP for this particular example. The verdict of this toy comparison is that TGP results in smaller, more economical trees if one is willing to pay the additional computational cost. This cost becomes prohibitively large for the more challenging examples that follow. Finally, let us mention that TGP is not able to model multi-output responses (a necessary attribute in order to deal with the examples that follow) and that the tree construction cannot be biased by the input probability distribution.



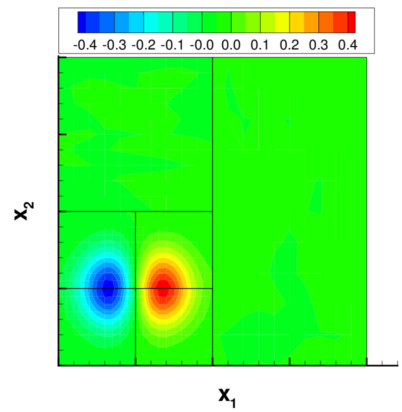
(a) MGP $N = 50$: tree and prediction.



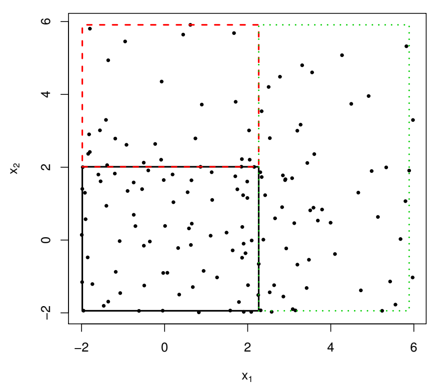
(b) MGP $N = 50$: observations.



(c) MGP $N = 70$: tree and prediction.



(d) MGP $N = 70$: observations



(e) TGP (ALM): tree and observations

Figure 2.2: Validation example: The MGP trees obtained for a tolerance $\delta = 10^{-3}$. In (a) $N = 50$. The 129 observations are shown in (b). In (c) $N = 70$. The 169 observations are shown in (d). Finally, in (e) we show the MAP tree obtained via the TGP model for a total of 150 samples gathered via ALM.

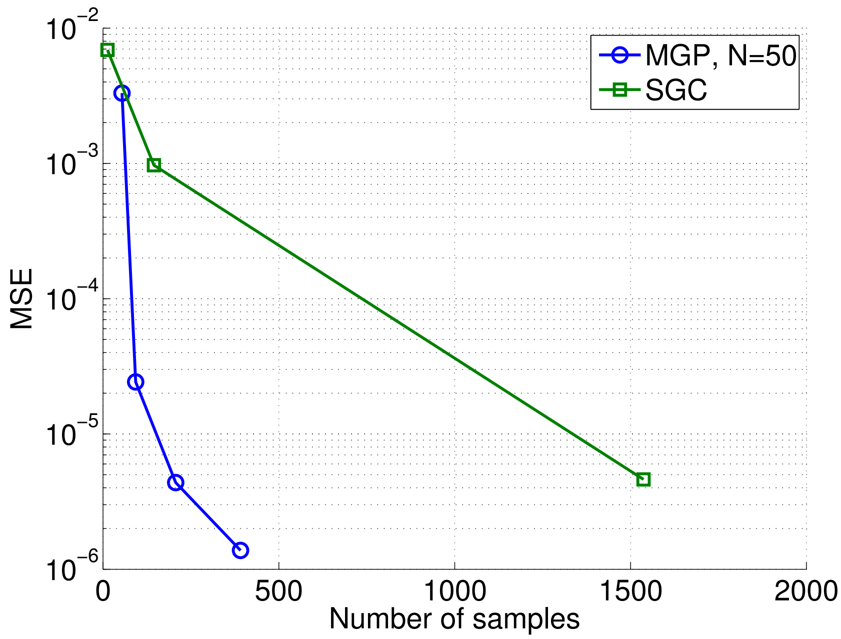


Figure 2.3: Validation example: The MSE in the prediction of $f(\mathbf{x})$ as a function of the observed samples for MGP and SGC. ASGC ($\epsilon = 10^{-3}$) is not reported since it fails to identify the localized features.

2.3.2 Krainchnan-Orszag three-mode problem

Consider the system of ordinary differential equations [91]:

$$
\begin{aligned}
\frac{dy_1}{dt} &= y_1 y_3, \\
\frac{dy_2}{dt} &= -y_2 y_3, \\
\frac{dy_3}{dt} &= -y_1^2 + y_2^2,
\end{aligned}
$$

subject to random initial conditions at $t = 0$. This dynamical system is particularly interesting because the response has a discontinuity at the planes $y_1(0) = 0$, $y_2(0) = 0$. The deterministic solver we use is a 4-th order Runge-Kutta method as implemented in GNU Scientific Library [30]. We solve the system for the time interval $[0, 10]$ and record the response at time step intervals of $\Delta t = 0.01$. This results in a total of $M = 300$ outputs (100 for each of the three dimensions of the response). We will consider two different cases of increasing difficulty with two and three input dimensions. For the two-dimensional case, we will also consider non-uniform input distributions. The results we obtain will be compared to a MC estimate with 10^6 samples. Let the MC mean and variance be $m_{r,\mathrm{MC}}$ and $v_{r,\mathrm{MC}}$, respectively, $r = 1, \dots, 300$. The error of the statistics will be evaluated using the (normalized) L_2 *norm of the error in variance* defined by:

$$
\mathrm{E}_{L_2} = \frac{1}{M} \sum_{r=1}^{M} \left(v_{r,\mathrm{MC}} - \mu_{v_r}\right)^2, \tag{2.63}
$$

where μ_{v_r} is the predictive mean of v_r (Equation 3.32). The results are compared with SGC and ASGC.

Two-dimensional Problem - Uniform Input For the two-dimensional problem, the stochastic initial conditions are defined by:

$$y_1(0) = 1,\ y_2(0) = 0.1x_1,\ y_3(0) = x_2,$$

where

$$x_i \sim U([-1, 1]),\ i = 1, 2.$$

This problem has a line discontinuity at $x_1 = 0$. We run the MGP framework for $N = 10$. Figure 2.4 shows the L_2 norm of the error in variance for MGP, SGC and ASGC as a function of the number of observations. At this example, the performance of MGP and ASGC ($\epsilon = 10^{-2}$) is approximately the same. Figure 2.5 depicts the prediction at y_3 ($t = 10$) along with the stochastic elements at levels of tolerance $\delta = 10^{-3}$, 10^{-5} and 10^{-7}. As a lower tolerance is reached, the stochastic mesh adapts around the discontinuity increasing the sampling density. Figure 2.6 plots the predictive mean and variance of $y_3(t)$ as a function of time t along with 95% error bars and compares it with the MC prediction. We notice that the error bars are over-estimated. Finally, by using 10^4 samples of the surrogate, we provide a kernel density approximation to the probability density functions (PDF) of y_2 ($t = 10$) and y_3 ($t = 10$) and compare them to the MC estimates with the same number of samples (Figure 2.7).

Two-dimensional Problem - Non-uniform Input We demonstrate the applicability of our method to non-uniform input distributions. As a first example, we consider a translated beta distribution:

$$p(x_i) \propto (2x_i - 1)^{a-1}(2x_i)^{b-1},\ i = 1, 2,$$

with $a = 2, b = 5$. This distribution assignes most of the probability mass to the bottom left corner of $[-1, 1]^2$. The left column of Figure 2.8 shows the de-

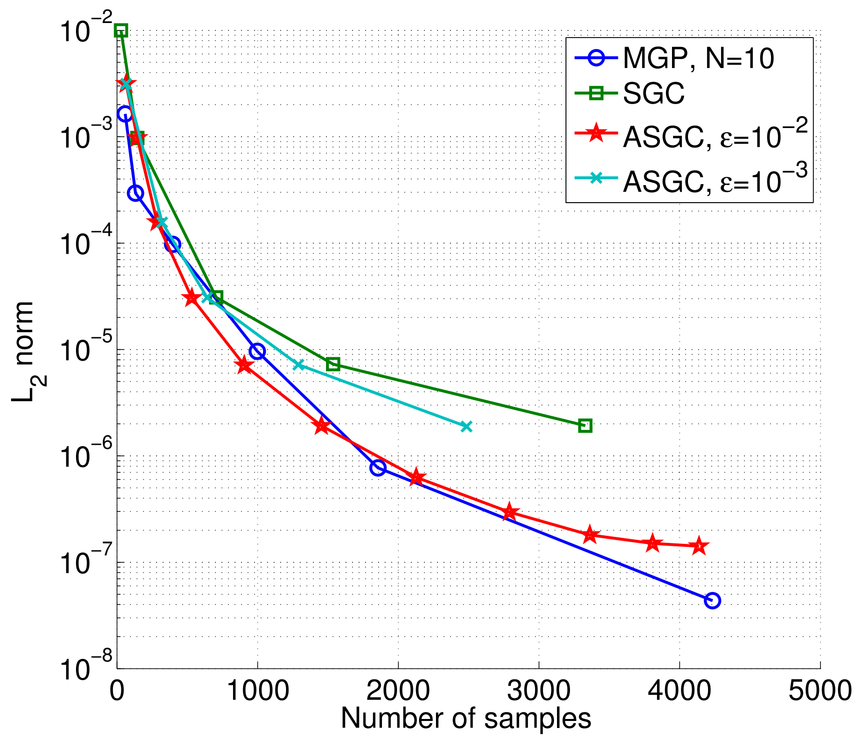


Figure 2.4: KO-2 (Uniform Input): the L_2 norm of the error in variance as a function of the observed samples for MGP, SGC and ASGC.

composition of the stochastic domain for various tolerances while the right column shows the observed inputs at each stage. It is important to notice that low probability regions are undersampled while the discontinuity is only partly resolved. Therefore, we can argue that the response surface is only resolved to the extent that it contributes to the statistics of the output. Finally, Figure 2.9 shows the observed inputs gathered at various tolerances for the case of a normal input distribution. The left column corresponds to a normal distribution with zero mean and unit variance while the left column corresponds to a normal distribution with mean $(-0.5, 0.5)$ and unit variance. Again we observe that unimportant regions are sampled less while the discontinuity is only resolved to the extent that it contributes to the statistics of the output.

Three-dimensional Problem - Uniform Input The three-dimensional problem is defined to have initial conditions:

$$y_1(0) = x_1,\ y_2(0) = x_2,\ y_3(0) = x_3,$$

where

$$x_i \sim U([-1, 1]), i = 1, 2, 3.$$

This case is notoriously difficult since it has multiple discontinuity planes. We run our framework for $N = 20$. Figure 2.10 shows the L_2 norm of the error in variance for MGP, SGC and ASGC as a function of the number of observations. ASGC with $\epsilon = 10^{-1}$ fails to converge, so it is not reported. Notice that MGP considerably out-performs ASGC. Figure 2.11 plots the predictive mean and variance of $y_3(t)$ as a function of time t along with 95% error bars and compares them with the corresponding MC predictions. Finally, Figure 2.12 plots the kernel density estimate of the PDF of y_2 $(t = 10)$ and of y_3 $(t = 10)$ using 10^4 samples of the surrogate.

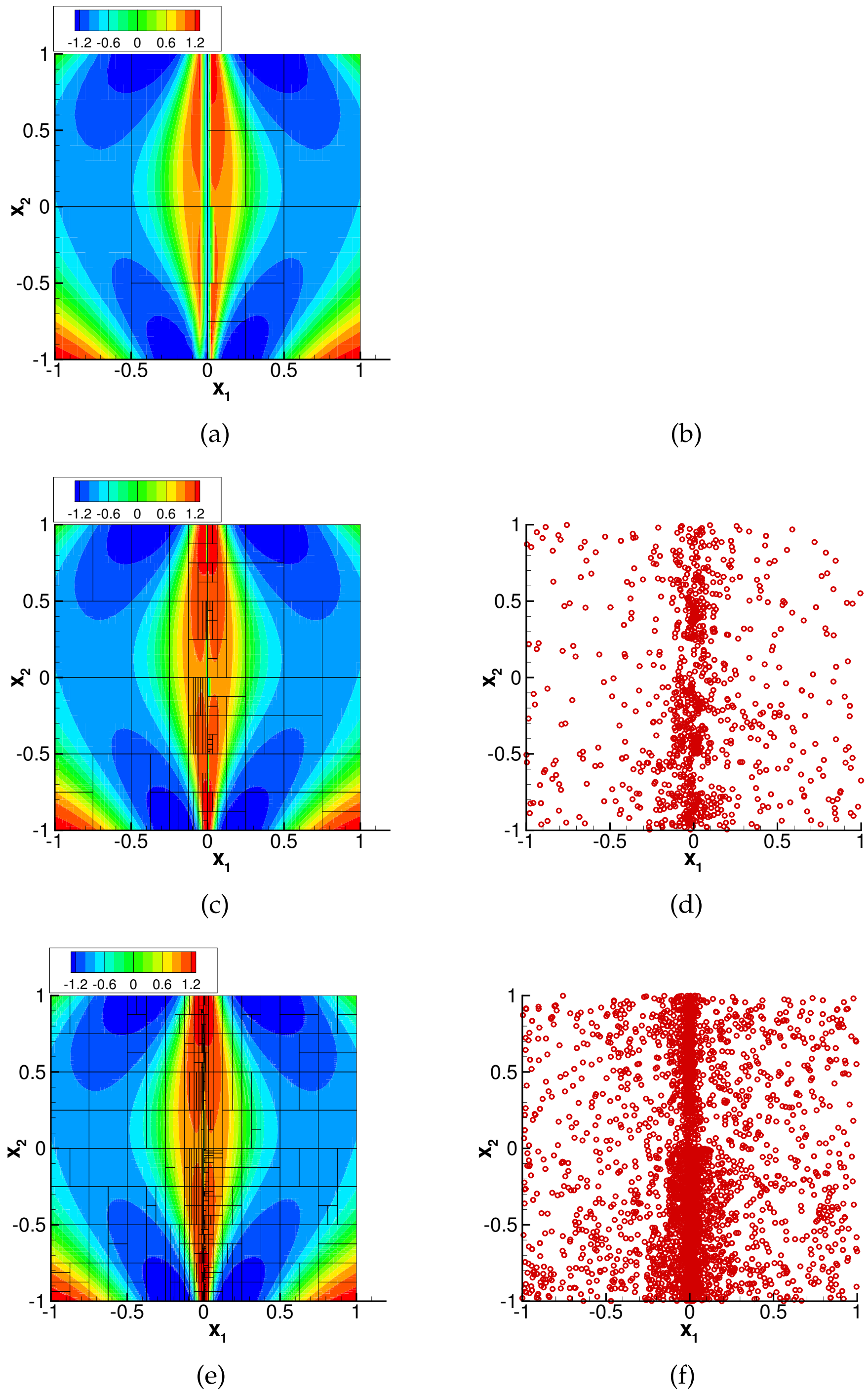


Figure 2.5: KO-2 (Uniform Input): The prediction at y_3 ($t = 10$) with the stochastic elements (left column, a, c, e) and the observed samples (right column, b, d, f) for tolerances (top to bottom) $\delta = 10^{-3}$, 10^{-5} and 10^{-7}.

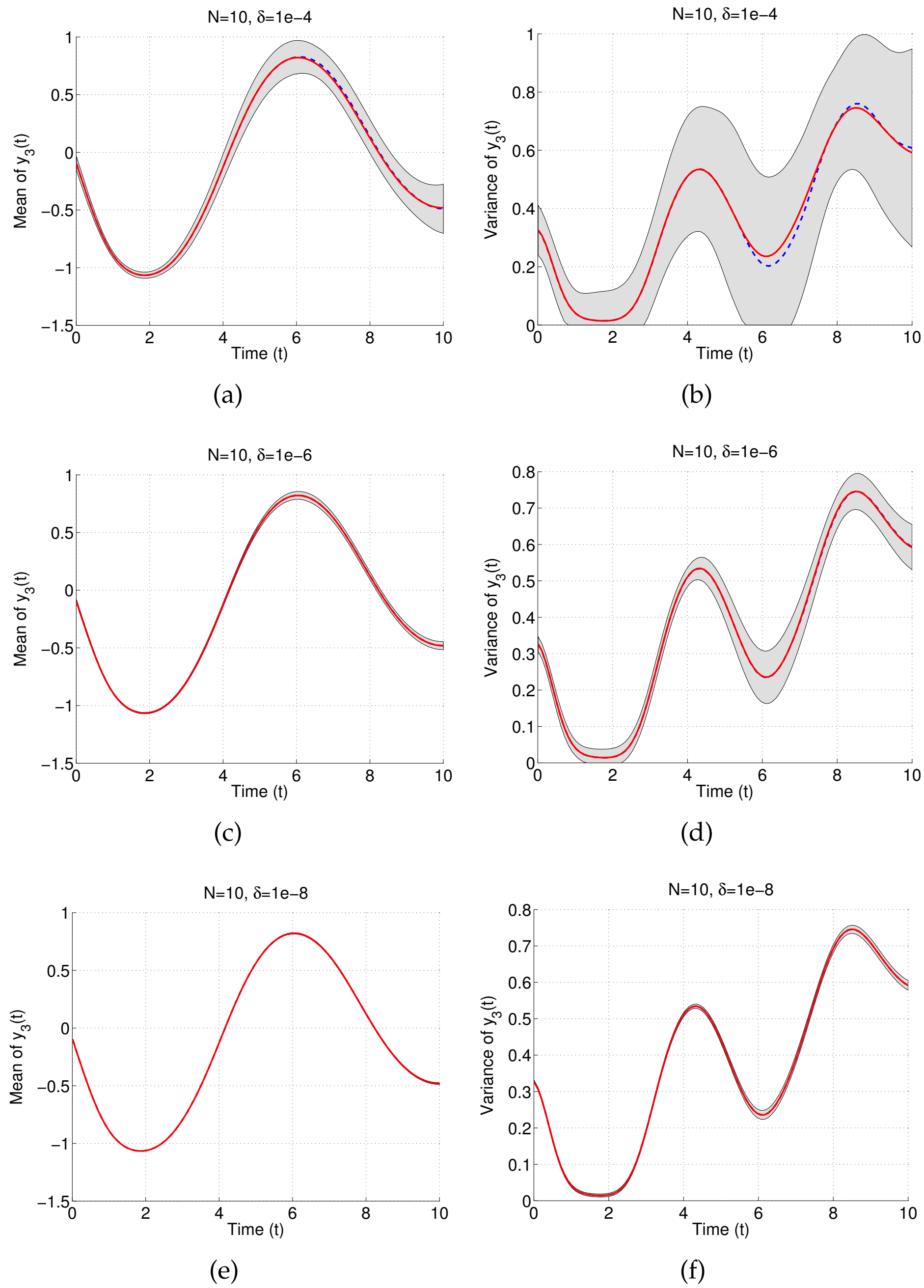
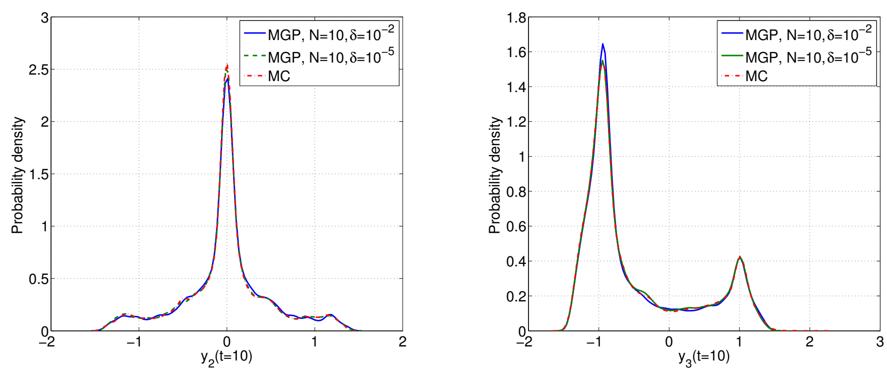


Figure 2.6: KO-2 (Uniform Input): predictive mean (dashed blue) versus MC estimate (solid red) of the mean (left column, a, c, e) and variance (right column, b, d, f) of $y_3(t)$ with 95% error bars for tolerances (top to bottom) $\delta = 10^{-4}, 10^{-6}$ and 10^{-8}.



(a) (b)

Figure 2.7: KO-2 (Uniform Input): kernel density estimation of the PDF of y_2 ($t = 10$) (left) and of y_3 ($t = 10$) (right) using 10^5 samples.

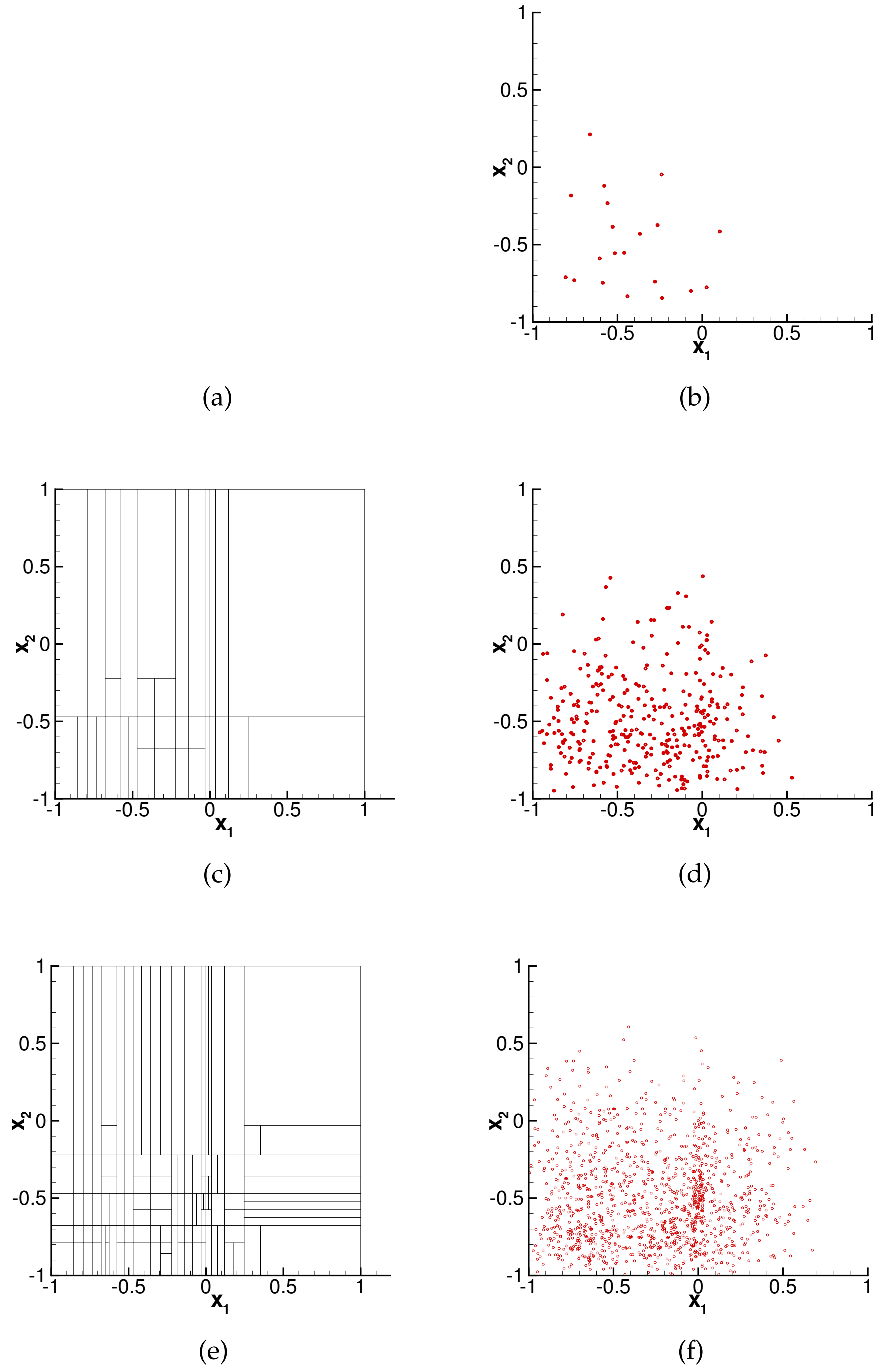
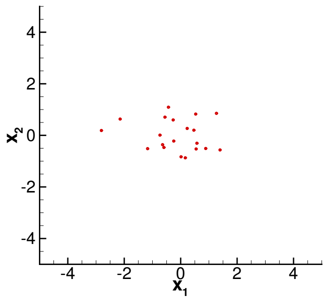
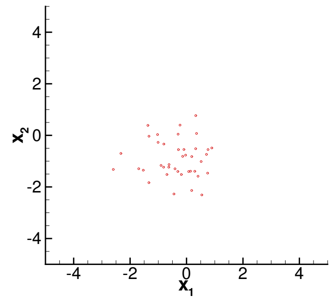


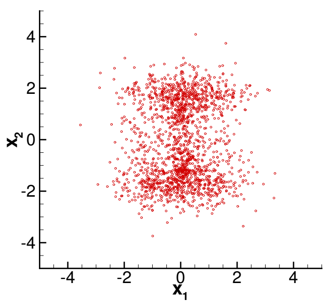
Figure 2.8: KO-2 (Beta input): The elements ((a), (c) and (e)) and the observed input points ((b), (d) and (f)) for beta input with $a = 2, b = 5$. The tolerances are (top to bottom) $\delta = 10^{-2}, 10^{-4}$ and 10^{-5}.



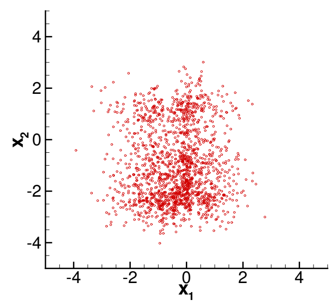
(a)



(b)



(c)



(d)

(e)

(f)

Figure 2.9: KO-2 (Normal input): The observed data for normal input distribution with unit variance and zero mean ((a), (c) and (e)) and mean $(-0.5, -0.5)$ ((b), (d) and (f)) The tolerances are (top to bottom) $\delta = 10^{-2}, 10^{-4}$ and 10^{-5}.

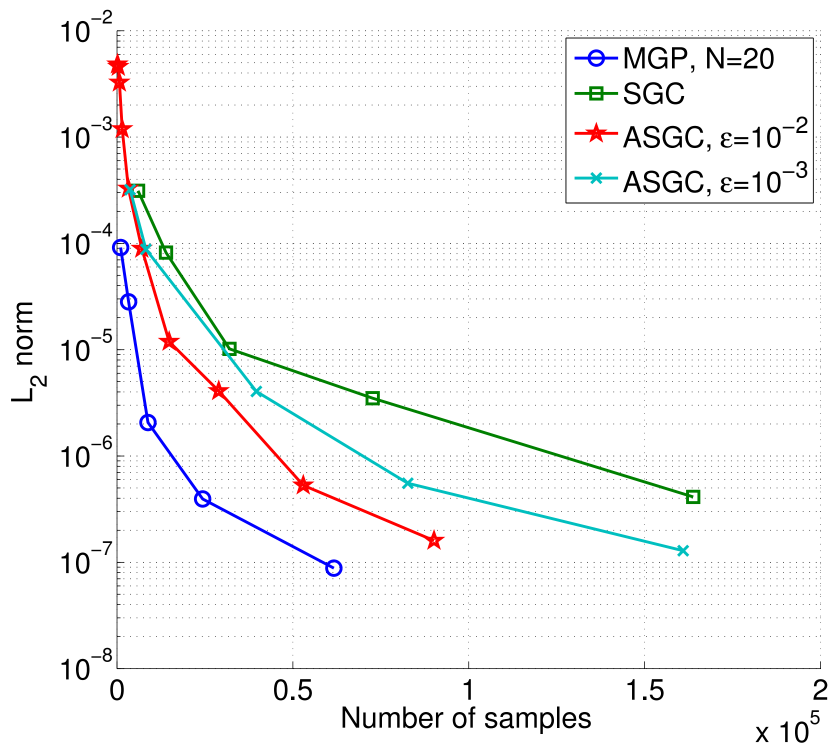


Figure 2.10: KO-3 (Uniform input): the L_2 norm of the error in variance as a function of the observed samples for MGP, SGC and ASGC.

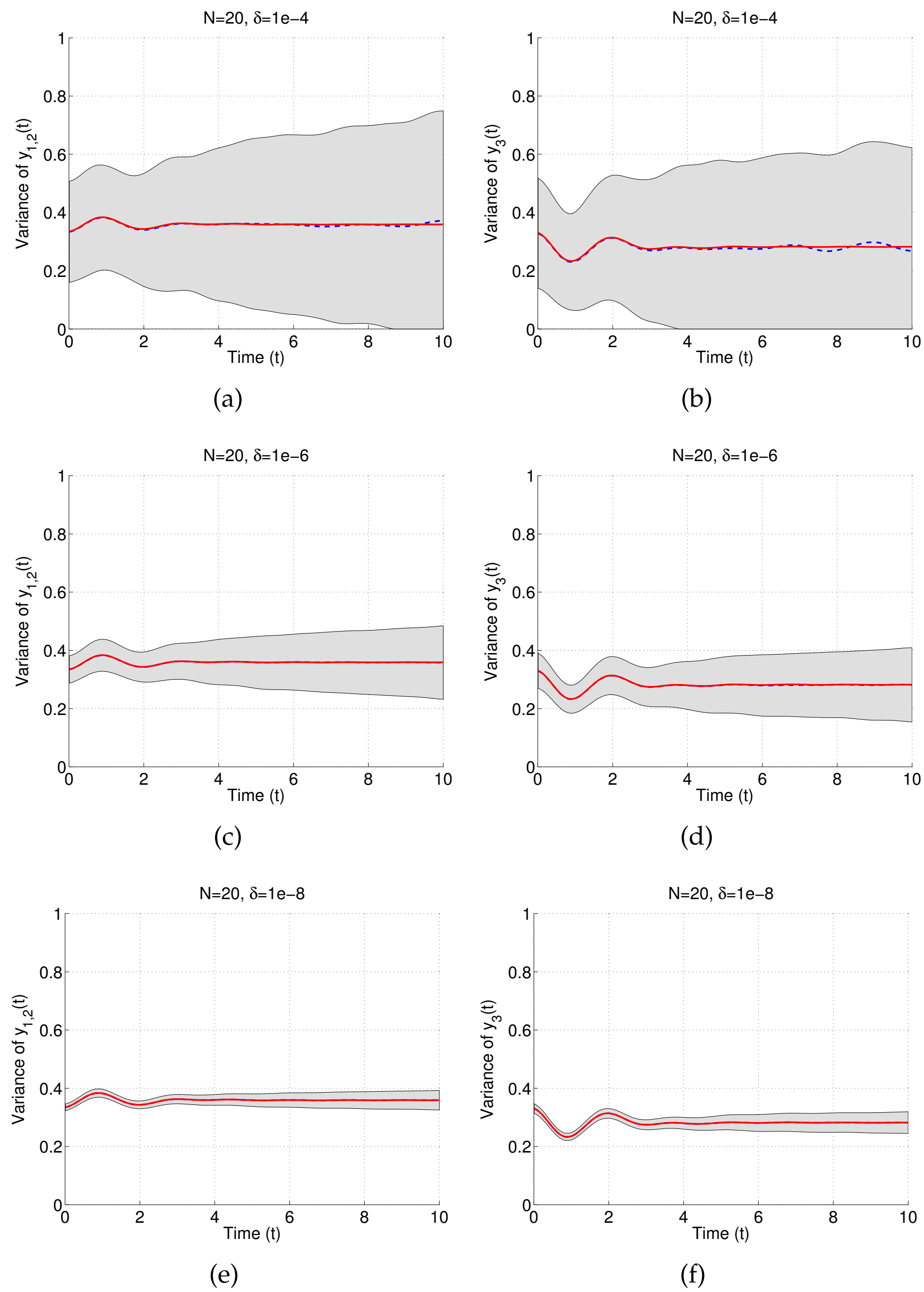
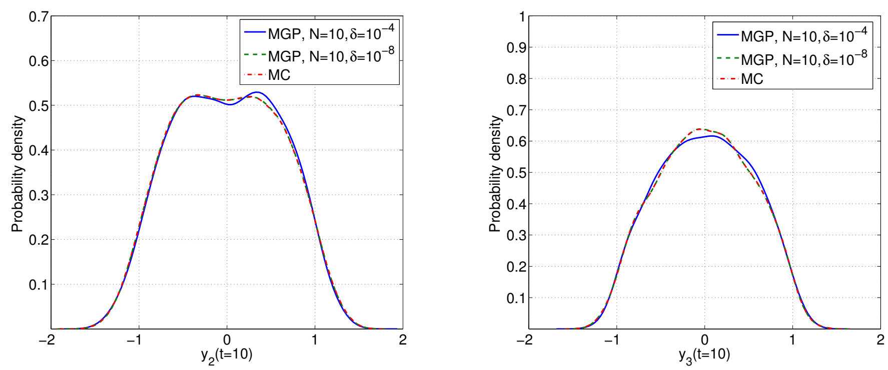


Figure 2.11: KO-3 (Uniform input): predictive mean (dashed blue) versus the MC estimate (solid red) of the variance of $y_1(t)$ and $y_2(t)$ (left column, a, c, e) and $y_3(t)$ (right column, b, d, f) with 95% error bars for tolerances (top to bottom) $\delta = 10^{-4}, 10^{-6}$ and 10^{-8}.



(a) (b)

Figure 2.12: KO-3 (Uniform input): kernel density estimation of the PDF of y_2 ($t = 10$) (left) and of y_3 ($t = 10$) (right) using 10^5 samples.

2.3.3 Elliptic Problem

In this section, we consider a simple stochastic elliptic problem [69]. Consider the stochastic partial differential equation (SPDE):

$$
\begin{aligned}
-\nabla \cdot (a_K(\boldsymbol{\omega}, \cdot)\nabla u(\boldsymbol{\omega}, \cdot)) &= f(\cdot), \quad \text{in } D, \\
u(\boldsymbol{\omega}, \cdot) = 0, \quad \text{on } \partial D,
\end{aligned}
$$

where the physical domain is $D = [0, 1]^2$. In order to avoid confusion with the physical dimension $\mathbf{x}$, we have chosen to denote the random variables with $\boldsymbol{\omega}$ instead of $\mathbf{x}$. We choose a smooth *deterministic* load:

$$
f(x, y) = 100 \cos(x) \sin(y),
$$

and work with homogeneous boundary conditions. The deterministic problem is solved with the finite element method using 400 (20×20 grid) bilinear quadrilateral elements. This results in $M = 441$ outputs (one for each node). The random diffusion coefficient $a_K(\boldsymbol{\omega}, x)$ is constructed to have a one-dimensional dependence:

$$
\log(a_K(\boldsymbol{\omega}, x, y) - 0.5) = 1 + \omega_1 \left(\frac{\sqrt{\pi} L}{2} \right)^{1/2} + \sum_{k=2}^{K} \xi_k \phi_k(x) \omega_k, \tag{2.64}
$$

where

$$
\xi_k := \left(\sqrt{\pi} L \right)^{1/2} \exp\left(\frac{-\left(\lfloor \frac{k}{2} \rfloor \pi L \right)^2}{8} \right), \quad \text{for } k \geq 2,
$$

and

$$
\phi_k(x) := \begin{cases} \sin\left(\frac{\lfloor \frac{k}{2} \rfloor \pi x}{L_p} \right) & \text{, if } k \text{ is even}, \\ \cos\left(\frac{\lfloor \frac{k}{2} \rfloor \pi x}{L_p} \right) & \text{, if } k \text{ is odd}, \end{cases}
$$

$\lfloor \cdot \rfloor$ being the integer part of real number. We choose the $\omega_k, k = 1, \ldots, K$ to be independent identically distributed random variables:

$$
\omega_k \sim U([-\sqrt{3}, \sqrt{3}]).
$$

Hence, the stochastic input space is $\mathbf{\Omega} = [-\sqrt{3},\ \sqrt{3}]^K$. Finally, we set:

$$L_p = \max\{1, 2L_c\} \text{ and } L = \frac{L_c}{L_p},$$

where L_c is called the *correlation length*. The expansion Equation 3.3.2 resembles the Karhunen-Loève expansion of a two-dimensional random field with stationary covariance

$$\mathrm{Cov}[\log(a_K - 0.5)]((x_1, y_1), (x_2, y_2)) = \exp\left(-\frac{(x_1 - x_2)^2}{L_c^2}\right).$$

In this study, we set the correlation length to $L_c = 0.6$ and test the convergence of our method for $K = 10, 20$ and 40 input dimensions. The results for $K = 10, 20$ and 40 are evaluated by calculating the L_2 error in variance (Equation 3.42) using a plain MC estimate with 10^6 samples. The performance is compared to ASGC for various ϵ. The $K = 10,\ 20$ and 40 cases are solved using $N = 20,\ 40$ and 80 up to a tolerance of 10^{-7}, 10^{-5} and 10^{-4}, respectively. Figures 2.13, 2.14 and 2.15 show the L_2 error in variance for each case. In all cases MGP outperforms ASGC, especially when the number of samples is small. For $K = 20$ and 40, the MC results can be trusted up to an accuracy of 10^{-3} in the L_2 norm. Therefore, the corresponding L_2 errors in variance for MGP and ASGC are shown in Figs. 2.14 and 2.15 up to the point that they become as accurate as MC. Figure 3.4 shows the convergence of the prediction for the variance of MGP as the tolerance threshold is lowered to $\delta = 10^{-7}$. Subfigure (e) of the same figure, plots the uncertainty of the variance $\sigma^2_{v_r}$ (Equation 2.44) at that tolerance. As already observed in previous examples, $\sigma^2_{v_r}$ over-estimates the true error. Fig. 2.17 tests the predictive capabilities of MGP for $K = 10$ at a tolerance $\delta = 10^{-6}$ on a random input point. We notice a good agreement with the true response.

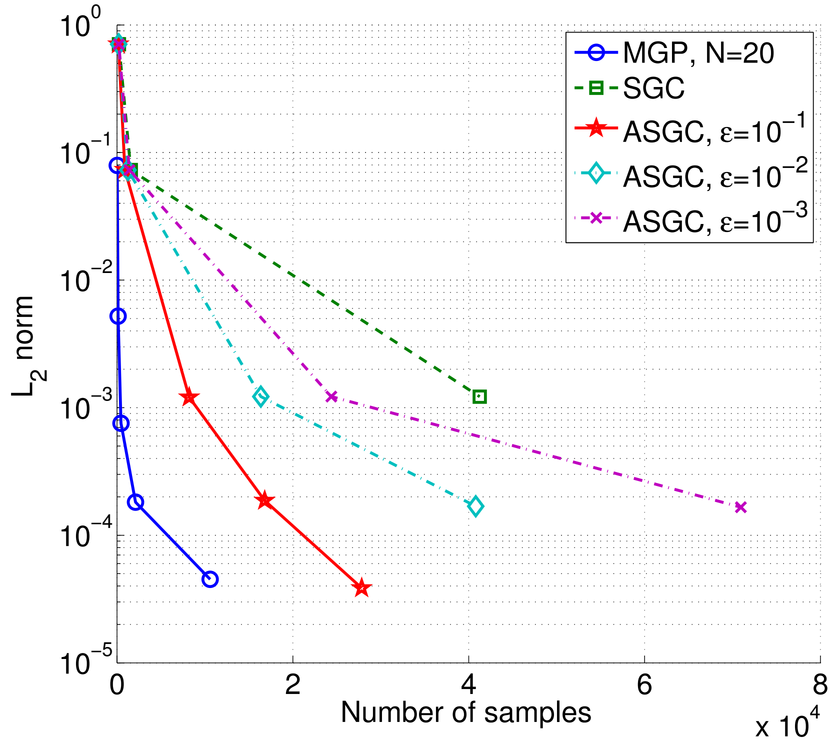


Figure 2.13: Elliptic, $K = 10$: The L_2 norm of the error in variance of the elliptic problem with $K = 10$ inputs as a function of the observed samples for MGP, SGC and ASGC.

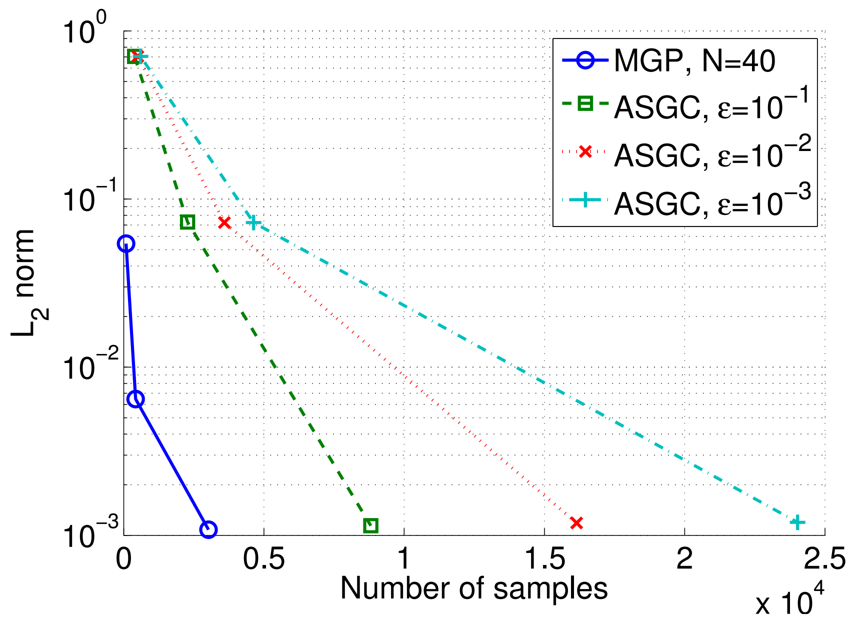


Figure 2.14: Elliptic, $K = 20$: The L_2 norm of the error in variance of the elliptic problem with $K = 20$ inputs as a function of the observed samples for MGP and ASGC.

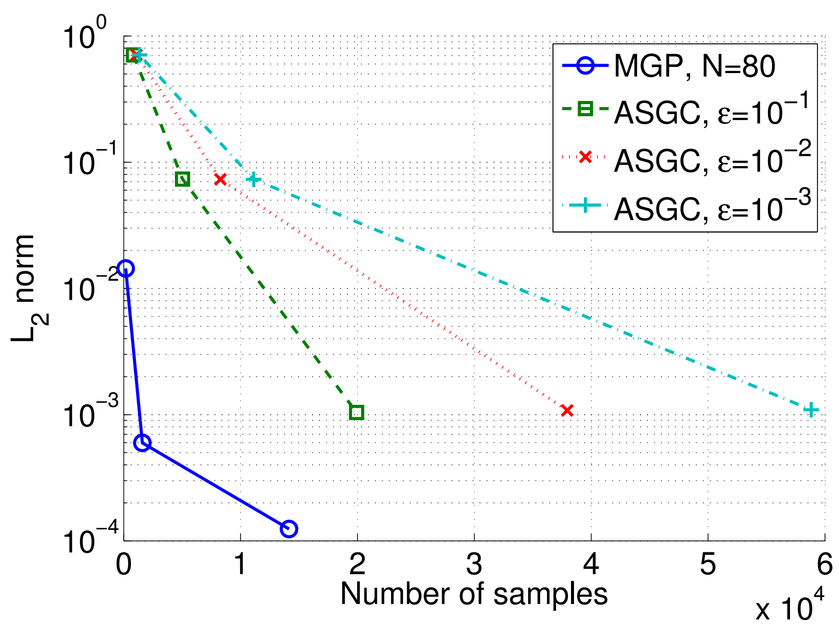


Figure 2.15: Elliptic, $K = 40$: The L_2 norm of the error in variance of the elliptic problem with $K = 40$ inputs as a function of the observed samples for MGP and ASGC.

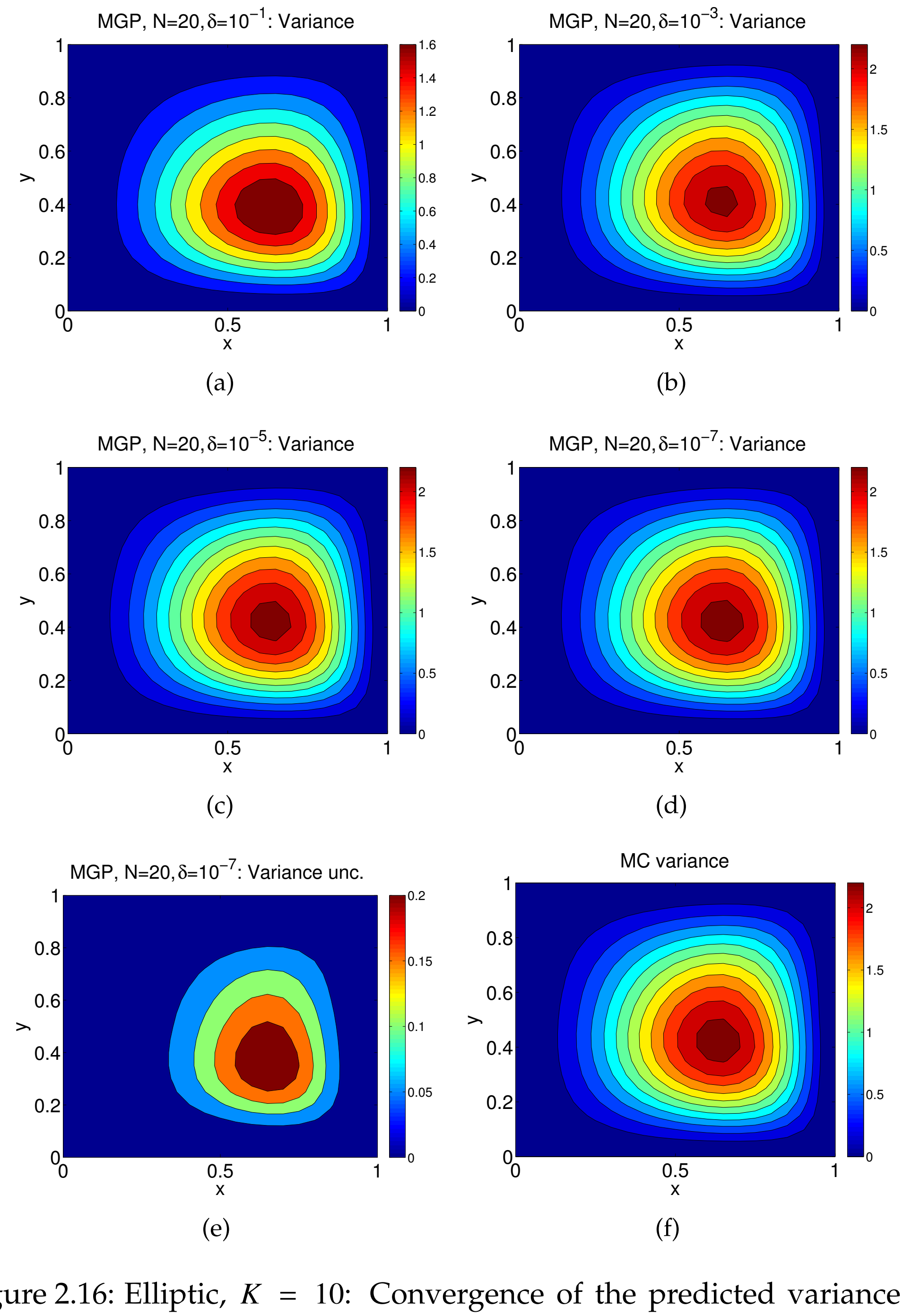


Figure 2.16: Elliptic, $K = 10$: Convergence of the predicted variance as the tolerance decreases. Subfigure (f) refers to MC results and subfigure (e) shows the uncertainty associated with the predicted variance $\sigma^2_{v_r}$ at $\delta = 10^{-7}$.

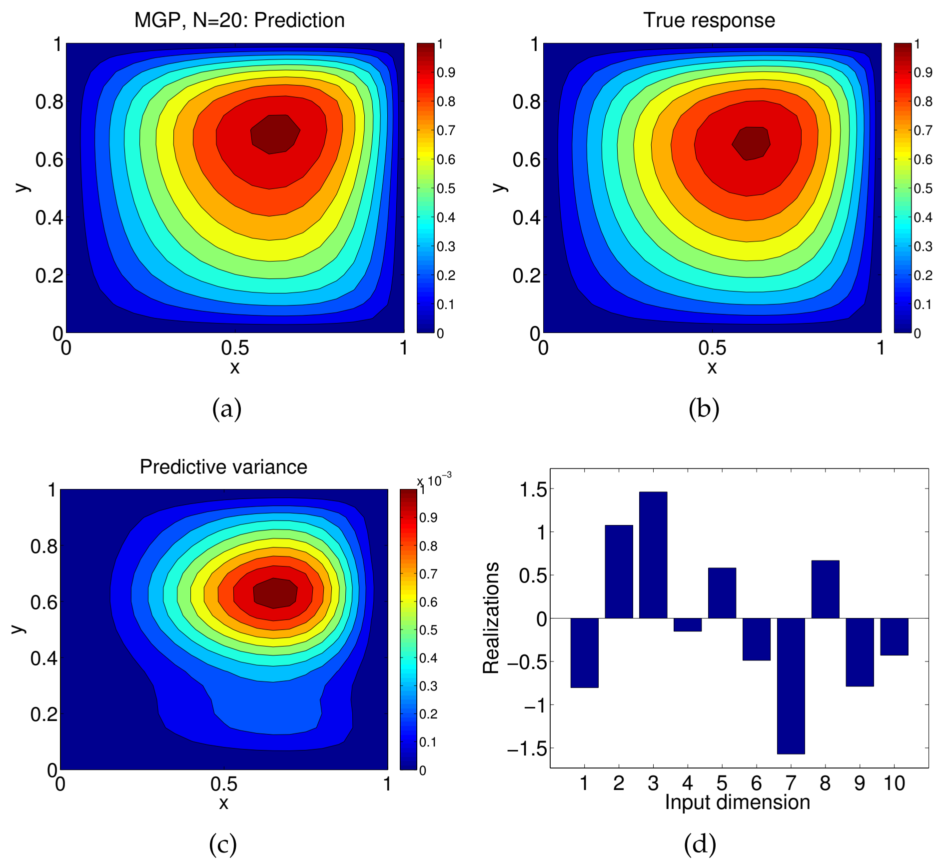


Figure 2.17: Elliptic, $K = 10, \delta = 10^{-6}$: Comparing the prediction (a) at a random input point (d) with the true response (b). Subfigure (c) shows the corresponding predictive variance.

2.3.4 Natural Convection Problem

Consider the dimensionless form of the Oberbeck-Boussinesq approximation using the vorticity transport equation in stream-function formulation:

$$
\begin{aligned}
-\frac{\partial}{\partial t}\nabla^2\psi - \frac{\partial\psi}{\partial y}\frac{\partial}{\partial x}\nabla^2\psi + \frac{\partial\psi}{\partial x}\frac{\partial}{\partial y}\nabla^2\psi &= -\,\mathrm{Pr}\,\nabla^4\psi + \mathrm{Ra}\,\mathrm{Pr}\,\frac{\partial T}{\partial x}, \qquad (2.65)\\
\frac{\partial T}{\partial t} + \frac{\partial\psi}{\partial y}\frac{\partial T}{\partial x} - \frac{\partial\psi}{\partial x}\frac{\partial T}{\partial y} &= \nabla^2 T, \qquad (2.66)
\end{aligned}
$$

where Pr and Ra are the Prandtl and Rayleigh numbers, respectively. In this formulation, the velocity field is given by:

$$
u = \frac{\partial\psi}{\partial y}, \quad v = -\frac{\partial\psi}{\partial x}. \qquad (2.67)
$$

We solve the problem in a two-dimensional square cavity $\mathbf{X} = [0,1]^2$. We impose no slip conditions to the boundary:

$$
u(x,y) = 0, \quad v(x,y) = 0, \text{ for } (x,y) \in \partial\mathbf{X}.
$$

The two horizontal walls are considered adiabatic:

$$
\frac{\partial T(x,y)}{\partial y} = 0, \text{ for } \; 0 \le x \le 1, y = 0, 1.
$$

The right vertical wall (hot) is kept at a constant temperature:

$$
T(1,y) = 0.5, \text{ for } \; 0 \le y \le 1.
$$

The left vertical wall (cold) is taken to be a one-dimensional Gaussian stochastic process with mean -0.5 and exponential covariance

$$
\mathrm{Cov}[x_1, x_2] = s^2 \exp\left\{-\frac{|x_1 - x_2|}{L_C}\right\},
$$

where s^2 is the variance of the signal and L_C the correlation length. Using the Karhunen-Loève (KL) expansion, we may write

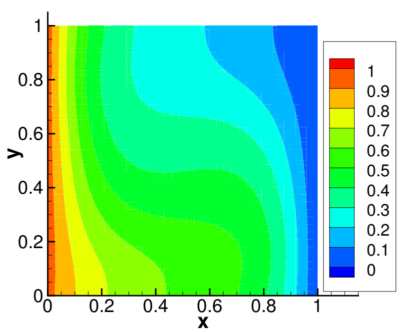
$$
T(0, y; \omega) = -0.5 + \sum_{k=1}^{\infty} \sqrt{\lambda_k}\phi_k(y)F^{-1}(\omega_k),
$$

where λ_k and $\phi_k(y)$ are the eigenvalues and eigenvectors of the covariance function and F^{-1} is the inverse cumulative distribution function of $\mathcal{N}(0, 1)$ and ω_k are independent uniform random variables in $[0, 1]$. It is noted here that λ_k and $\phi_k(y)$ are analytically available [97].

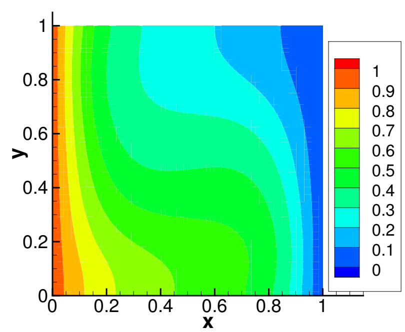
In this study, we set $L_C = 1$ and keep only $K = 4$ or 8 terms in the KL expansion. The parameters we use are $\mathrm{Pr} = 1$ and $\mathrm{Ra} = 5000$. The deterministic problem is solved using the Nektar fluid dynamics code [**?**], which utilizes spectral elements. The domain was decomposed in 240 quadrilateral elements (12×12 grid) and 4 spectral modes were used on each one. It has been numerically verified that no more modes were necessary for convergence of the spectral elements. The output is observed at 16 (4×4 grid) equidistant mesh points on each element. This results in a total of 2401 outputs for each of the physical quantities of interest (T, u, v and the pressure p). The total number of output dimensions is thus $M = 9604$. For computational convenience, we only work with temperature T and the u component of the velocity, a total of 4802 output parameters. For $K = 2$ and 4, we run our scheme until a tolerance $\delta = 10^{-5}$ is reached with $N = 10$. A total of 1393 and 14396 observations were made, respectively. For $K = 8$, we reach a tolerance of $\delta = 10^{-3}$ which results in 829 observations being made. Fig. 2.18 compares the predicted standard deviations (std.) of u (top) and T (bottom) for $K = 8$ with MC estimates using $80,000$ samples. The results are in good agreement with the MC estimates. In Figs. 2.19, we draw a random sample from the input distribution for the $K = 4$ case. We present the predictive mean of T along with two std.'s and compare it to the absolute error. Notice that the two std.'s are qualitatively similar to the absolute error of the prediction.

(a) K=8, MGP ($N = 20, \delta = 10^{-3}$): std. of u

(b) K=8, MC: std. of u

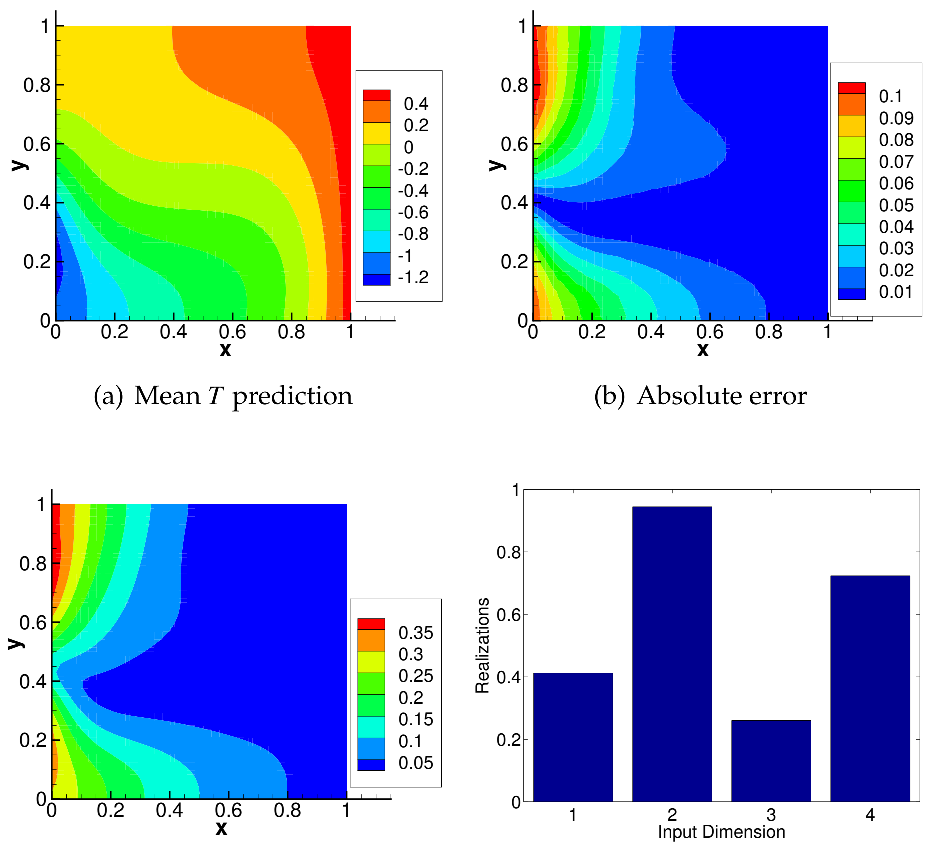


(c) K=8, MGP ($N = 20, \delta = 10^{-3}$): std. of T



(d) K=8, MC: std. of T

Figure 2.18: Natural Convection: MGP prediction at tolerance level $\delta = 10^{-3}$ for the standard deviation of the velocity u (top) and temperature T (bottom) compared to a MC estimate for $K = 8$ input dimensions.



(a) Mean T prediction

(b) Absolute error

(c) 2 std.'s of T

(d) Input point

Figure 2.19: Natural Convection ($K = 4, \delta = 10^{-5}$): Comparing the prediction at a random input point with the true response.

2.4 Conclusions

We have developed a novel, non-intrusive Bayesian scheme based on a treed multi-output GP model that can be used in UQ tasks. The tree is built in a sequential way that utilizes information contained only in the data observed so far. Tree refinement depends on the observations through a global measure of the uncertainty in the prediction, the inferred length scales as well as the input probability distribution. A Sequential Experimental Design technique based on the predictive uncertainty was also used to adaptively select the most informative input points on each element. The final result is a non-stationary, predictive distribution for the response of the underlying system, that can be semi-analytically integrated to provide point estimates and error bars for the statistics of interest. We have numerically demonstrated that the framework can (1) capture non-stationary responses, (2) locate discontinuities, (3) identify localized features and (4) reduce the sampling frequency on unimportant input dimensions. The method was shown to outperform SGC and ASGC in almost all numerical examples investigated, especially when only a small number of observations were used.

The presented framework is particularly interesting, in that it can be extended in several ways that can improve its performance dramatically. From a technical point of view several aspects require further numerical investigation: e.g. the dependence of the result on the choice of the maximum number of samples per element N, the performance of the ALC experimental design technique instead of the ALM scheme used in the current work, the dependence of the final decomposition of the stochastic space on the refinement criterion Equation 3.38 for $q \neq 1$ and so on. Another important development would be to re-place the current multi-output GP model with a GP model that explicitly takes into account correlation between the outputs. Such an effort, is expected to reduce the number of samples required significantly. Currently, the GPs learnt on each element are dropped if the element is split in half. The result is that each element is treated independently and the response is not smooth along the element boundaries. Alternatively, another treed GP model can be formulated in which the children of a node would learn the residual of the response instead of the response itself. In such a way, the upper nodes of the tree would model coarse features of the response, while localized features would be resolved by the leaves of the tree. Finally, a great deal of effort must be put in mathematically working out the error bounds in the various statistics that result from the uncertainty of the prediction. As already mentioned in Section 2.2.2, the proper Bayesian way to account for the uncertainty of the predicted statistics, would be via an MC procedure: we would sample a complete response surface from the full model, integrate it with respect to the input probability distribution and obtain a sample of the statistics. The mathematical details of such a procedure are the subject of our current research.

CHAPTER 3

RELEVANCE VECTOR MACHINES

3.1 Introduction

In Chapter 2 we built a tree surrogate of Gaussian processes (GPs). The aforementioned work used attributes specific to GPs for the adaptation criteria. In this chapter, we extend our tree construction methodology to arbitrary input distributions and arbitrary local Bayesian regressions. As a replacement of GPs, we develop a multi-output version of the Relevance Vector Machine (RVM) [88], which we call Multi-output RVM (MRVM). RVM is a Bayesian sparse kernel technique for regression and classification that shares many characteristics with the well-established Support Vector Machines (SVM) [20]. Section 7.2 of [7] discusses the similarities and differences between SVM and RVM. Let us briefly mention that RVM diminishes several of the difficulties present in SVM, for example (1) one does not have to use cross-validation techniques in order to choose the complexity parameters and (2) the basis functions employed in regression can be arbitrary instead of strictly positive definite kernels.

The beginning of Section 3.2 provides some basic definitions and assumptions. In Section 3.2.1, we introduce the MRVM model for arbitrary local basis functions and in Section 3.2.2, we provide fast training algorithm based on the evidence approximation. Various derivations and specific details of the constituents parts of the algorithm can be found in Appendix A. In Section 3.2.3, we make a specific choice for the basis functions, namely local square exponential kernels centered on top of each observed input point and locally defined set of orthogonal polynomials. Sections 3.2.4 and 3.2.5 discuss how the local and global statistics, respectively can be evaluated, while in Section 3.2.6 we devise a scheme that allows us to sample from the predictive distribution of the statistics. In Section 3.2.7, we introduce our tree construction methodology that is largely independent of the specifics of the local Bayesian regression model. The method is demonstrated numerically for various UQ problems in Section 3.3. Finally, we conclude in Section 3.4.

3.2 Methodology

Let $\mathbf{X}$ represent the parameter space of a physical model and $p(\mathbf{x})$ a probability density function (PDF) defined on $\mathbf{X}$. $\mathbf{X}$ will be referred to as the *stochastic input space*. We assume that the stochastic problem has been formulated in such a way that $\mathbf{X}$ is a rectangle of $\mathbb{R}^K$ for some $K \geq 1$, that is $\mathbf{X} = \times_{k=1}^{K}[a_k, b_k]$, with $-\infty \leq a_k < b_k \leq +\infty$ the upper and lower bounds of each dimension. Furthermore, we suppose that all dimensions of $\mathbf{X}$ are independent. This is just a convenient assumption present in many other UQ methodologies. If this assumption does not hold, then one would have to transform the input so that it does. Such a transformation always exists [80]. Finding it, however, is beyond the scope of the present work. Thus, we may write $p(\mathbf{x}) = \prod_{k=1}^{K} p_k(x_k)$, where p_k is the PDF pertaining to the k-th input dimension.

Let us now consider the multi-output function $\mathbf{f} : \mathbf{X} \to \mathbb{R}^M$ representing the result of a computer code (deterministic solver) modelling a physical system, i.e. at a given input point $\mathbf{x} \in \mathbf{X}$, the predicted response of the system is $\mathbf{f}(\mathbf{x})$. We will write

$$\mathbf{f} = (f_1, \ldots, f_M),$$

and refer to f_r as the r-th output of the response function, $r = 1, \ldots, M$. In this work, we will identify $\mathbf{f}(\mathbf{x})$ as the true response of an underlying physical system and we will ignore any modelling errors. The input probability distribution induces a probability distribution on the output. The UQ problem involves the calculation of the statistics of the output $\mathbf{y} = \mathbf{f}(\mathbf{x})$. Quantities of particular interest are the *q-moments* $\mathbf{m}^q = (m_1^q, \ldots, m_M^q)$, defined for $q \geq 1$ and $r = 1, \ldots, M$ by:

$$m_r^q := \int_{\mathbf{X}} f_r^q(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}, \tag{3.1}$$

as well as functions of them. In particular, the *mean* $\mathbf{m} = (m_1, \ldots, m_M)$:

$$m_r := m_r^1 = \int_{\mathbf{X}} f_r(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}, \tag{3.2}$$

and the *variance* $\mathbf{v} = (v_1, \ldots, v_M)$:

$$v_r := \int_{\mathbf{X}} (f_r(\mathbf{x}) - m_r)^2 \, p(\mathbf{x}) d\mathbf{x} = m_r^2 - (m_r^1)^2. \tag{3.3}$$

The statistics will be calculated by interrogating a surrogate of $\mathbf{f} : \mathbf{X} \to \mathbb{R}^M$. This surrogate will be put together from local surrogates defined over $I \geq 1$ *elements* of the stochastic space $\mathbf{X}^i \subset \mathbf{X}$ such that

$$\mathbf{X} = \cup_{i=1}^I \mathbf{X}^i \text{ and } \operatorname{int}(\mathbf{X}^i) \cap \operatorname{int}(\mathbf{X}^j) = \emptyset, \forall i, j \in I, i \neq j, \tag{3.4}$$

where $\operatorname{int}(\mathbf{X}^i)$ denotes the interior of the set $\mathbf{X}^i$ under the usual Euclidean metric of $\mathbb{R}^K$. The response surface is correspondingly decomposed as

$$\mathbf{f}(\mathbf{x}) := \sum_{i=1}^I \mathbf{f}^i(\mathbf{x}) 1_{\mathbf{X}^i}(\mathbf{x}), \tag{3.5}$$

where $1_{\mathbf{X}^i}(\mathbf{x})$ is the indicator function of $\mathbf{X}^i$ and $\mathbf{f}^i(\cdot)$ is the restriction of $\mathbf{f}(\cdot)$ on $\mathbf{X}^i$. The local surrogates will be identified as Multi-output Relevant Vector Machines (MRVM) defined over the stochastic element $\mathbf{X}^i$. These MRVMs will be trained by observing $\mathbf{f}^i(\cdot)$. The MRVM model is outlined in Section 3.2.1 and a training algorithm is provided in Section 3.2.2. The predictive mean of the MRVMs will be used to derive semi-analytic estimates of all moments $\mathbf{m}^q$ (Sections 3.2.4 and 3.2.5). An addendum of the Bayesian treatment, is the ability to provide error bars for the point estimates of the moments (Section 3.2.6). This feature is absent from most current UQ methods. Our aim is to create a surrogate by making as few calls to the computer program as possible. This is achieved by adaptively decomposing the domain (tree construction) based on the predicted variability of the response function as well as biasing by the underlying input probability density $p(\mathbf{x})$ (Section 3.2.7).

3.2.1 Multi-output Relevance Vector Machine

We turn our focus to a single element of the stochastic space $\mathbf{X}^i \subset \mathbf{X}$ and discuss the construction of a local surrogate model based on some already observed data. The choice of the elements is the subject of Section 3.2.7. All quantities introduced herein are local to the element $\mathbf{X}^i$. However, in order to avoid having an unnecessarily cumbersome notation, we do not explicitly show this dependence. We assume that we have observed a fixed number $N \geq 1$ of data points

$$\mathcal{D} := \left\{(\mathbf{x}^{(n)}, \mathbf{y}^{(n)} = \mathbf{f}\left(\mathbf{x}^{(n)}\right))\right\}_{n=1}^{N}, \ \mathbf{x}^{(n)} \sim p^i(\mathbf{x}), \tag{3.6}$$

where $p^i(\mathbf{x})$ is the conditional PDF on $\mathbf{X}^i$, defined by:

$$p^i(\mathbf{x}) = \frac{p(\mathbf{x})}{P\left(\mathbf{X}^i\right)} 1_{\mathbf{X}^i}(\mathbf{x}), \tag{3.7}$$

and $P\left(\mathbf{X}^i\right)$ is the probability that a random input point falls in $\mathbf{X}^i$:

$$P\left(\mathbf{X}^i\right) := \int_{\mathbf{X}^i} p(\mathbf{x}) d\mathbf{x}. \tag{3.8}$$

Because we wish to model all outputs simultaneously, it is necessary to scale them so that they exhibit the same signal strength. Towards this end, let us introduce the *observed means*:

$$\mu_{\mathrm{obs},r} = \frac{1}{N}\sum_{n=1}^{N} y_r^{(n)}, \tag{3.9}$$

and the *observed variances*:

$$\sigma^2_{\mathrm{obs},r} = \frac{1}{N-1}\sum_{n=1}^{N}(y_r - \mu_{\mathrm{obs},r})^2, \tag{3.10}$$

for $r = 1,\dots,M$, of the data $\mathcal{D}$. We will be modeling the *scaled response functions* $g_r : \mathbf{X}^i \to \mathbb{R}$, defined by

$$g_r(\mathbf{x}) = \frac{f_r(\mathbf{x}) - \mu_{\mathrm{obs},r}}{\sigma_{\mathrm{obs},r}}, r = 1,\dots,M. \tag{3.11}$$

Obviously, this definition depends on the actual observations. However, we expect that if N is big or if the stochastic element under investigation is small, then it is a good approximation to the ideal scaling, i.e. zero mean and unit variance for all outputs.

We model the mean of each g_r as

$$\mu_{g_r}(\mathbf{x}) = \mathbf{w}_r^T\boldsymbol{\phi}(\mathbf{x}), \tag{3.12}$$

where $S \geq 1$, $\mathbf{w}_r = (w_{r1},\dots,w_{rS})^T$ is the vector of the weights for the r-th output and $\boldsymbol{\phi}(\mathbf{x}) = (\phi_1(\mathbf{x}),\dots,\phi_S(\mathbf{x}))^T$ are some basis functions defined over the element $\mathbf{X}^i$. At this point, let us introduce the scaled version of the observations $\mathcal{D}_{\mathrm{sc}} = \{(\mathbf{x}^{(n)},\mathbf{z}^{(n)})\}_{n=1}^N$, where $\mathbf{z}^{(n)} = \left(z_1^{(n)},\dots,z_M^{(n)}\right)^T$ with $\mathbf{z}_r^{(n)} = \frac{y_r^{(n)}-\mu_{\mathrm{obs},r}}{\sigma_{\mathrm{obs},r}}$, and let $\mathcal{D}_{\mathrm{sc},r} = \{(\mathbf{x}^{(n)}, z_r^{(n)})\}_{n=1}^N$ be the observations in $\mathcal{D}_{\mathrm{sc}}$ that pertain to the r-th output dimension. We assume that, given the weights, the scaled observations $z_r = g_r(\mathbf{x})$ are normally distributed about $\mu_{g_r}(\mathbf{x})$ with inverse variance $\beta > 0$, i.e.

$$p(z_r|\mathbf{x},\mathbf{w}_r,\beta) = \mathcal{N}\left(z_r|\mu_{g_r}(\mathbf{x}),\beta^{-1}\right), r = 1,\dots,M. \tag{3.13}$$

Under the assumption that the added noise is independent for each sample point, the *likelihood* of the scaled data $\mathcal{D}_{\mathrm{sc},r}$ related to the r-th output can be written as:

$$p(\mathcal{D}_{\mathrm{sc},r}|\mathbf{w}_r,\beta) = \prod_{n=1}^{N} p(z_r^{(n)}|\mathbf{x}^{(n)},\mathbf{w}_r,\beta), \tag{3.14}$$

and assuming the output dimensions are conditionally independent given the weights, the likelihood of all the scaled observed data $\mathcal{D}_{\mathrm{sc}}$ is:

$$p(\mathcal{D}_{\mathrm{sc}}|\mathbf{W},\beta) = \prod_{r=1}^{M} p(\mathcal{D}_{\mathrm{sc},r}|\mathbf{w}_r,\beta), \tag{3.15}$$

where $\mathbf{W}$ is the $M \times S$ matrix whose rows are given by $\mathbf{w}_r$. We impose a *prior* probability density on each weight w_{rs} of the form:

$$p(w_{rs}|\alpha_s) = (2\pi)^{-1/2}\alpha_s^{1/2}\exp\left\{-\frac{\alpha_s w_{rs}^2}{2}\right\},\ \ s = 1,\dots,S, \tag{3.16}$$

where α_s, $s = 1,\dots,S$ are unknown *positive* hyper-parameters (one for each basis function). We will collectively denote those by $\boldsymbol{\alpha} = (\alpha_1,\dots,\alpha_S)^T$. Notice that all output dimensions $r = 1,\dots,M$ share the same $\boldsymbol{\alpha}$. Furthermore, we assume that the weights of the r-th output are conditionally independent given $\boldsymbol{\alpha}$:

$$p(\mathbf{w}_r|\boldsymbol{\alpha}) = \prod_{s=1}^{S} p(w_{rs}|\alpha_s),$$

as well as that all the $\mathbf{w}_r$'s are conditionally independent given $\boldsymbol{\alpha}$:

$$p(\mathbf{W}|\boldsymbol{\alpha}) = \prod_{r=1}^{M} p(\mathbf{w}_r|\boldsymbol{\alpha}).$$

Following [88], it is easy to show using matrix identities, that the *posterior distribution* of the weights $\mathbf{w}_r$ is given by

$$p(\mathbf{w}_r|\mathcal{D}_{\mathrm{sc},r},\boldsymbol{\alpha},\beta) = \mathcal{N}(\mathbf{w}_r|\boldsymbol{\mu}_r,\boldsymbol{\Sigma}), \tag{3.17}$$

where

$$\boldsymbol{\Sigma} = \left(\mathrm{diag}(\boldsymbol{\alpha}) + \beta\boldsymbol{\Phi}^T\boldsymbol{\Phi}\right)^{-1} \quad \text{and} \quad \boldsymbol{\mu}_r = \beta\boldsymbol{\Sigma}\boldsymbol{\Phi}^T\mathbf{z}_r, \tag{3.18}$$

with $\mathrm{diag}(\boldsymbol{\alpha})$ being a diagonal matrix with $\boldsymbol{\alpha}$ at the diagonal, $\boldsymbol{\Phi} = \left[\boldsymbol{\phi}\left(\mathbf{x}^{(1)}\right)\ldots\boldsymbol{\phi}\left(\mathbf{x}^{(N)}\right)\right]^T$ is the $N \times S$ *design matrix*, and $\mathbf{z}_r = \left(z_r^{(1)}, \ldots, z_r^{(N)}\right)^T$ the scaled version of the observed r-th outputs. Under the *evidence approximation* (see Ch. 3 of [7]), point estimates of $\boldsymbol{\alpha}$ and β can be found by maximizing the *marginal likelihood* defined by

$$p(\mathcal{D}_{\mathrm{sc}}|\boldsymbol{\alpha},\beta) = \int p(\mathcal{D}_{\mathrm{sc}}|\mathbf{W},\beta)p(\mathbf{W}|\boldsymbol{\alpha})d\mathbf{W}. \tag{3.19}$$

We will calculate this integral analytically and provide an efficient algorithm for its maximization in Section 3.2.2.

To conclude this section, let us give the predictive distribution of our model. Let $\boldsymbol{\alpha}$ and β be the hyperparameters that maximize Equation 3.19. From [88] and scaling back to the original function $f_r(\mathbf{x})$, it is easy to show that the predictive distribution is:

$$p(y_r|\mathbf{x},\mathcal{D}) = \mathcal{N}(y_r|\mu_{f_r}(\mathbf{x}), \sigma^2_{f_r}(\mathbf{x})), \tag{3.20}$$

where the *predictive mean* is given by:

$$\mu_{f_r}(\mathbf{x}) = \sigma_{\mathrm{obs},r}\boldsymbol{\mu}_r^T\boldsymbol{\phi}(\mathbf{x}) + \mu_{\mathrm{obs},r} \tag{3.21}$$

and the *predictive variance* by:

$$\sigma^2_{f_r}(\mathbf{x}) = \sigma^2_{\mathrm{obs},r}\left(\beta^{-1} + \boldsymbol{\phi}(\mathbf{x})^T\boldsymbol{\Sigma}\boldsymbol{\phi}(\mathbf{x})\right). \tag{3.22}$$

Remark 4 *Treating the outputs jointly. One can argue that treating the outputs completely independently (i.e. each output having its own set of hyper-parameters) would raise the need for scaling them and would also increase the flexibility of the model. The main drawback of such an approach is that inference of the hyper-parameters has to be carried out as many times as the number of outputs times the number of elements. In high-dimensional output scenarios, this approach becomes computationally intense. On* *the other hand, the joint treatment presented in this section requires the inference of the hyper-parameters only once per element. We have chosen the joint approach mainly on the grounds that it is computationally efficient. In realistic applications, the outputs can be separated in groups according to their nature and a different MRVM can be used on each group. Extending our methodology to this case is straightforward.*

Remark 5 *Posterior independence of the weights. As observed above, for each given output $r = 1, \dots, M$, the vector of weights $\mathbf{w}_r$ has non-diagonal covariance given by $\mathbf{\Sigma}$. On the other hand, the vectors of weights corresponding to different outputs are a posteriori independent. This is expected, since we have made no attempt to capture the correlation between the various outputs. On one hand, this is a drawback of the proposed model since it will definitely lead to sub-optimal use of the available observations. On the other hand, it is a silently made assumption in practically all UQ methodologies with which we are familiar. We have chosen to address this issue in our work in [4].*

3.2.2 Maximization of the Marginal Likelihood

We develop a maximization framework for the marginal likelihood only to the hyper-parameters α and β. In [89], it is shown that the logarithm of the marginal likelihood of the r-th output is given by

$$\mathcal{L}(\boldsymbol{\alpha}|\mathcal{D}_{\mathrm{sc},r}) := \log p(\mathcal{D}_{\mathrm{sc},r}|\boldsymbol{\alpha},\beta) = -\frac{1}{2}\left[N\log 2\pi + \log|\mathbf{C}| + \mathbf{z}_r^T\mathbf{C}^{-1}\mathbf{z}_r\right], \tag{3.23}$$

where the $N \times N$ matrix $\mathbf{C}$ is defined by:

$$\mathbf{C} = \beta^{-1}\mathbf{I} + \mathbf{\Phi}\,\mathrm{diag}(\boldsymbol{\alpha})^{-1}\mathbf{\Phi}^T \tag{3.24}$$

with $\mathbf{I}$ being the identity matrix and $|\cdot|$ the determinant operator. Under the assumptions of the previous section, the logarithm of the marginal likelihood of all the data $\mathcal{L}(\boldsymbol{\alpha},\beta|\mathcal{D}_{\mathrm{sc}})$, is given by the sum of $\mathcal{L}(\boldsymbol{\alpha},\beta|\mathcal{D}_{\mathrm{sc},r})$ for $r = 1,\ldots,M$. We will be working with a normalized version of $\mathcal{L}(\boldsymbol{\alpha},\beta|\mathcal{D}_{\mathrm{sc}})$, which we call the *evidence*:

$$\begin{aligned} \mathcal{E}(\boldsymbol{\alpha}|\mathcal{D}_{sc}) &:= \frac{1}{MN}\mathcal{L}(\boldsymbol{\alpha},\beta|\mathcal{D}_{\mathrm{sc}}) \\ &= -\frac{1}{2}\log 2\pi - \frac{1}{2N}\log|\mathbf{C}| - \frac{1}{2MN}\sum_{r=1}^{M}\mathbf{z}_r^T\mathbf{C}^{-1}\mathbf{z}_r. \end{aligned} \tag{3.25}$$

The evaluation of this quantity is based on the Generalized Singular Value Decomposition (GSVD) [37] of the matrices $\mathbf{A} = \beta^{1/2}\,\mathrm{diag}(\boldsymbol{\alpha})^{1/2}$ and $\boldsymbol{\Phi}$ and is discussed in Appendix C.2. Fix the hyper-parameters $\boldsymbol{\alpha}$ and let $\boldsymbol{\alpha}_{-s}$ denote the S-dimensional vector with $\alpha_s = +\infty$, that is with the s-th basis function removed. An important observation about the evidence $\mathcal{E}(\boldsymbol{\alpha})$ is that it can be decomposed in two terms (see Appendix A.1):

$$\mathcal{E}(\boldsymbol{\alpha}) = \mathcal{E}(\boldsymbol{\alpha}_{-s}) + \epsilon(\alpha_s), \tag{3.26}$$

where $\mathcal{E}(\boldsymbol{\alpha}_{-s})$ is the evidence of the model without the s-th basis function and $\epsilon(\alpha_s)$ (Equation A.1) depends only on α_s and the sufficient statistics h_s and q_{rs}, $s = 1,\ldots,S, r = 1,\ldots,M$ defined in Equation A.2. A study of the stationary points of $\epsilon(\alpha_s)$ subject to $\alpha_s > 0$, is carried out in Appendix A.2 and reveals that there exist two distinct possibilities depending on the value of a statistic θ_s (Equation A.6): (1) If $\theta_s > 0$, there exists a finite $\alpha_s^{\mathrm{new}} > 0$ that is a unique global maximum given (Equation A.7); (2) If $\theta_s \leq 0$, $\epsilon(\alpha_s)$ is maximized at $\alpha_s = +\infty$, that is when the s-th basis function is removed. This observation suggests an iterative algorithmic procedure that is guaranteed to increase the evidence at each step converging to a local maximum. We start with a single basis function, for example randomly chosen, setting $\alpha_s = +\infty$ for the rest. At each step, there are three possible actions: (1) Add a new basis function; (2) Re-estimate the hyper-parameters of an existing basis function; and (3) Remove a basis function. The action that results in the maximum change in evidence is selected. In Appendix A.3 we explicitly discuss each possible action and how to calculate the change in evidence.

As is noted also by other authors [39], the choice of β can have a critical impact on the predictive capabilities of a computer surrogate. Since β corresponds to the inverse noise of the model, one would expect it to be relatively large indicating the very low (if existent) noise of computer experiments. However, in the case of limited data and/or an inadequate basis set (e.g. polynomials of a given degree), a very big β might lead to severe over-fitting. In these cases, including a finite amount of noise penalizes too complex models and can improve the predictive performance of the surrogate. A natural way to choose β is to maximize the evidence with respect to it also. Optimizing the evidence jointly for $\boldsymbol{\alpha}$ and β would make the model computationally intractable. For this reason, we have devised the following heuristic that results in a local maximum of the evidence. We first optimize the evidence with respect to $\boldsymbol{\alpha}$ for a fixed β with the procedure outlined in the previous paragraph. Then we keep $\boldsymbol{\alpha}$ fixed and maximize the evidence with respect to β. The latter optimization involves a function of a single variable and can be easily carried out by utilizing a golden section based algorithm. We iterate between the two optimizations until a desired tolerance in the evidence has been reached. Algorithm 1 outlines the maximization procedure. For further details on how to evaluate the various quantities that are involved, you may consult Appendix C.2.

Step 1: Initialize $\boldsymbol{\alpha}$ by including the single basis function maximizing the evidence (i.e. all α_s's are infinity except for one) and pick a starting value for β (we use $\beta = 100$ in all numerical experiments).
Step 2: Compute all the statistics (H_m, Q_{rm}), (h_m, q_{rm}) and θ_m, $s = 1, \ldots, S$ and $r = 1, \ldots, M$, using Eqs. (A.3), (A.4) and (A.6), respectively.
Step 3: Iterate through all basis functions and calculate the change in evidence that results from their optimal action (add, re-estimate and remove) as discussed in Appendix A.3. Select the basis function that results in the maximum change in evidence.
Step 4: If the maximum change in evidence is less than a threshold (in the numerical examples we use 10^{-6}), then go to Step 5. Otherwise, perform the selected action and go to Step 2.
Step 5: Find the maximum of the evidence with respect to β while keeping $\boldsymbol{\alpha}$ fixed. If the change in evidence is less than a threshold, then stop. Otherwise, update the value of β and go to Step 2.

Algorithm 2: MRVM training procedure.

3.2.3 On the choice of the basis functions

The framework developed thus far is independent of the choice of the basis function $\phi_s(\mathbf{x})$. The only requirement is that each element $\mathbf{X}^i$ has its own set of locally defined basis functions. In the numerical examples section, we investigate the performances of two possible choices:

1. Squared Exponential Kernels: Here, we choose $\phi_s(\mathbf{x})$ to be kernel functions centered on top of the observed data points. Assume that we are working on element $\mathbf{X}^i$ and that we have observed $\mathcal{D}$ as in Equation 3.6. Then, we define $\phi_s(\mathbf{x})$ to be $\phi_s(\mathbf{x}) := \phi(\mathbf{x}, \mathbf{x}^{(s)}), s = 1, \ldots, N$, where $\phi(\cdot, \cdot)$ is a kernel function. That is, $S = N$. A usual choice of $\phi(\cdot, \cdot)$ is the Square Exponential (SE) kernel $\phi(\mathbf{x}_1, \mathbf{x}_2) = \exp\left\{-\frac{1}{2}\sum_{k=1}^{K}\frac{(x_{1k}-x_{2k})^2}{\ell_k^2}\right\}$. The parameters ℓ_k have the interpretation of the length scale of each particular dimension. They can be selected by a cross validation procedure or by maximizing the evidence

with respect to them also. To keep our framework as simple as possible, we fix the length scales on each element to be proportional to the standard deviation of the observed input dimensions, i.e. $\ell_k := \left(\frac{1}{N}\sum_{n=1}^{N}(x_k^{(n)} - \bar{x}_k)^2\right)^{\frac{1}{2}}$, where $\bar{x}_k = \frac{1}{N}\sum_{n=1}^{N} x_k^{(n)}$.

2. Optimal Orthogonal Polynomials: One can choose the $\phi_s(\mathbf{x})$'s as in [92] to be the locally defined set of optimal polynomials with respect to the conditional density $p^i(\mathbf{x})$ of the element $\mathbf{X}^i$. By the independence assumption, $p^i(\mathbf{x}) = \prod_{k=1}^{K} p_k^i(x_k)$, where $p_k^i(x_k)$ is the conditional distribution pertaining to the k-th dimension. Using the Lanczos procedure of [32], we construct the one-dimensional orthogonal polynomials with respect to $p_k^i(x_k)$ for all $k = 1, \ldots, K$ and then multiply them to obtain K-variate orthogonal polynomials with respect to $p^i(\mathbf{x})$ up to a given degree. In that case, the RVM framework is used to decide how many and which of these polynomials should be kept. As will be discussed in Section 3.2.4, the optimal orthogonal polynomials have the nice of property of yielding analytic estimates for the mean and the variance without the need to fit the second power of the response.

3.2.4 Calculation of the local statistics

As before, the focus is again on a specific element $\mathbf{X}^i$. All quantities are again local to $\mathbf{X}^i$. In order to keep notational complexity to a minimum, we do not explicitly show this dependence. We will derive point estimates for the mean and the higher moments of the response based on the linear point estimator of $\mathbf{f}$ over $\mathbf{X}^i$ given in Equation 3.21. To be exact, we are interested in estimating all (local) moments $\mathbf{m}^q = (m_1^q, \ldots, m_M^q)$, where

$$m_r^q = \int_{\mathbf{X}^i} f_r^q(\mathbf{x}) p^i(\mathbf{x}) d\mathbf{x}, \tag{3.27}$$

and $q \geq 1$ and $p^i(\mathbf{x})$ is the conditional probability distribution of $\mathbf{X}^i$ given in Equation 3.7. For the case of non-orthogonal basis functions (i.e. Square Exponentials), we can derive semi-analytic estimates by keeping concurrent MRVM estimates of the response raised to the q power. In particular, the q power of the response is treated also as an MRVM with its own hyper-parameters $\boldsymbol{\alpha}^q$. Let us denote the predictive mean for the q power of the response at $\mathbf{x} \in \mathbf{X}^i$ by:

$$\mu_{f_r^q}(\mathbf{x}; \boldsymbol{\alpha}^q) = \sigma^q_{\mathrm{obs},r} \left(\boldsymbol{\mu}_r^q\right)^T \boldsymbol{\phi}(\mathbf{x}) + \mu^q_{\mathrm{obs},r},$$

where $\mu^q_{\mathrm{obs},r}$ and $\sigma^q_{\mathrm{obs},r}$ are defined as in Eqs. (3.9) and (3.10), respectively, using the q power of the observed response and $\boldsymbol{\mu}^q$ is the posterior mean of the weights of the q power of the scaled response (see Equation 3.18). The q-moment can be approximated by:

$$\hat{m}_r^q = \int_{\mathbf{X}^i} \mu_{f_r^q}(\mathbf{x}; \boldsymbol{\alpha}^q) p^i(\mathbf{x}) d\mathbf{x}. \tag{3.28}$$

Fortunately, the integrals involved can be expressed in terms of expectations of the basis functions with respect to the conditional input distribution. Due to the independence assumption of the input random variables and the special choice of basis functions made in Section 3.2.3, those expectations can be numerically evaluated using a quadrature rule[1] in $O(K)$ time (K being the number of input dimensions) for special choices of basis functions $\phi_s(\mathbf{x})$. This is possible for the exponential kernels chosen in this work (see Section 3.2.3) and for any polynomial basis. In particular, for the exponential kernels it is straightforward to

[1] In our numerical examples, we used the QAGS algorithm of QUADPACK as implemented in GSL. This is based on an adaptive bisection scheme combined with the Wynn epsilon-algorithm that utilizes a Gauss-Kronrod 21-point rule (see [75]). This algorithm is obviously an overkill for the task under consideration, but its cost is negligible compared to a single run of the deterministic code.

show that:

$$\hat{m}_r^q = \sigma_{\text{obs},r}^q \sum_{s=1}^{S} \left[\mu_{rs}^q \prod_{k=1}^{K} \left(\int_{a_k^i}^{b_k^i} \exp\left\{ -\frac{\left(x_k - x_k^{(s)}\right)^2}{2\ell_k^2} \right\} p_k^i(x_k) dx_k \right) \right] + \mu_{\text{obs},r}^q, \tag{3.29}$$

where $p_k^i(x_k)$ is the marginal conditional distribution of $\mathbf{X}^i = \times_{k=1}^K [a_k^i, b_k^i]$ $(-\infty \leq a_k^i < x_k < b_k^i \leq +\infty)$ along dimension k:

$$p_k^i(\mathbf{x}) := \int_{\times_{\ell \neq k}[a_\ell^i, b_\ell^i]} p^i(\mathbf{x}) dx_1, \dots, dx_{k-1} dx_{k+1} \dots dx_K = \frac{p_k(x_k) 1_{[a_k^i, b_k^i]}(x_k)}{\int_{a_k^i}^{b_k^i} p_k(x_k) dx_k}. \tag{3.30}$$

The $S \times K$ one-dimensional integrals are the ones that are evaluated numerically. In case the support of $p_k^i(x_k)$ is infinite (or semi-infinite), the integrand is mapped - using a suitable transformation - to $[-1, 1]$. For arbitrary basis functions, one would have to use a MC based integration technique, which would also be computationally efficient since the evaluation of the local surrogates is extremely cheap compared to the underlying deterministic solver.

When optimal orthogonal polynomials are used as basis functions, then it is trivial to derive analytic estimates for the moments by exploiting their orthogonality [92] without requiring an estimate of the second power of the response. Assuming they are normalized, the first two moments are given by:

$$\hat{m}_r{}^1 = \sigma_{\text{obs},r}^1 \mu_{r1}^1 + \mu_{\text{obs},r}^1, \text{ and } \hat{m}_r{}^2 = (\sigma_{\text{obs,r}}^1)^2 \sum_{s=1}^{S} (\mu_{rs}^1)^2,$$

where μ_{r1}^1 is the coefficient of the constant polynomial.

3.2.5 From local to global statistics

In the same spirit as the multi-element methods [91, 92, 28], we combine the statistics over each stochastic element in order to obtain the global analogues.

Since we now work over the whole domain, we will explicitly mark the dependence of the underlying quantities on the element $\mathbf{X}^i, i = 1, \dots I$. Let $\hat{m}_r^{q,i}$ be the estimate for the local q-moment that pertains to the element $\mathbf{X}^i$. An estimate of the global q-moment is provided by:

$$\hat{m}_r^q = \sum_{i=1}^{I} \hat{m}_r^{q,i} P(\mathbf{X}^i), \tag{3.31}$$

where $P\left(\mathbf{X}^i\right)$ was defined in Equation 3.8 and can be easily be numerically evaluated due to the independence assumption. Finally, an estimate of the variance of the response is obtained via:

$$\hat{v}_r = \hat{m}_r^2 - \left(\hat{m}_r^1\right)^2. \tag{3.32}$$

3.2.6 Quantifying the uncertainty of the predicted statistics

The Bayesian nature of the underlying regression model, induces a probability distribution on the space of possible surrogates and this, in turn, on the predicted statistics. In order to make this connection clear, one may think of the space of possible surrogates as being parametrized by the weights. Then, the posterior of the weights $\mathbf{w}_r$ of each element (see Equation 3.17) spawns a posterior distribution on the space of surrogates. The weight of this posterior is directly associated with the epistemic uncertainty due to the limited number of observations. In this section, we describe a procedure that yields samples from the posterior distribution of the surrogates. Given a sample surrogate we can compute its statistics, quantifying thereby the impact that the epistemic uncertainty has on them. In what follows, we work within a specific element $\mathbf{X}^i$ even though this is not explicitly denoted in order to limit the notational burden.

Let $\boldsymbol{\alpha}$ denote the hyper-parameters and $p(\mathbf{w}_r|\mathcal{D}_{\text{sc},r}, \boldsymbol{\alpha})$ the posterior of the weights, as implied by Equation 3.17. We obtain a sample of the q-th moment in the following way:

1. Sample $\mathbf{w}_{\text{sample},r}$ from $p(\mathbf{w}_r|\mathcal{D}_{\text{sc},r}, \boldsymbol{\alpha})$ for $r = 1, \dots, M$ (Equation 3.17). This is a normal disribution and can be sampled directly.
2. Then, a sample of $f_r(\mathbf{x}), r = 1, \dots, M$ is given by:

$$f_{\text{sample},r}(\mathbf{x}) = \sigma_{\text{obs},r} \left(\mathbf{w}_{\text{sample},r}\right)^T \boldsymbol{\phi}(\mathbf{x}) + \mu_{\text{obs},r}. \tag{3.33}$$

3. Finally, use Equation 3.33 to obtain a sample of the local q-th moment:

$$\hat{m}^q_{\text{sample},r} = \int_{\mathbf{X}^i} f^q_{\text{sample},r}(\mathbf{x}) p^i(\mathbf{x}) d\mathbf{x}. \tag{3.34}$$

In order to obtain a sample of the global q-th moment, the above mentioned procedure is repeated on each element and the results are combined in the spirit of Section 3.2.5. For the case of optimal orthogonal polynomials, Step 3 can be carried out analytically as discussed in Section 3.2.4. For the case of the SE basis functions, Step 3 has to be carried out numerically. If one is also fitting the q-th power of the response in order to make use of the semi-analytic formulas of Equation 3.29, then she has to modify the above procedure so that it samples the posterior of the weights $\mathbf{w}^q_r$ pertaining to that particular power.

3.2.7 Adaptivity

In this section, we develop an iterative procedure to adaptively decompose the stochastic space in small elements. The initial step of this procedure starts by considering a single element, that is the whole stochastic space $\mathbf{X}$. Here, we assume that we are already given a decomposition of the domain as well as a local surrogate model on each element. The decision we wish to make is whether or not to refine a given element and in which way. We develop refinement criteria that are based solely on information gathered by the current surrogate model and no further calls to the deterministic solver are required. The Bayesian predictive variance Equation 3.22 is used to define a measure of our uncertainty about the prediction over the whole domain $\mathbf{X}$. We show how this measure can be broken down to contributions coming from each element. Based on this observation, we derive a criterion that suggests refinement of an element if its contribution to the global uncertainty is larger than a pre-specified threshold. For the sake of simplicity, we only consider rectangular elements and refine them by splitting them perpendicular to the dimension of greater importance. The importance of a particular dimension is characterized by the variability of the predicted response function and biased by the underlying input probability distribution.

Suppose that we have already a decomposition of the stochastic domain $\mathbf{X}$ in *rectangular* elements $\mathbf{X}^i$, e.g. $\mathbf{X}^i = [a_1^i, b_1^i] \times \cdots \times [a_K^i, b_K^i]$, with $-\infty \le a_k^i < b_k^i \le +\infty, k = 1, \dots, K, i = 1, \dots, I$ such that Equation 3.4 holds. Furthermore, assume that we have already learnt the local surrogates on each element $\mathbf{X}^i$. Let $\sigma^2_{f_r^i}(\mathbf{x})$ be the predictive variance of the r-th output of the local surrogate of $\mathbf{f}^i$ at $\mathbf{x} \in \mathbf{X}^i$ (Equation 3.22). By the conditional independence assumption for the predictive distribution over each element and Equation 3.5, the predictive variance of the r-th output of the global surrogate $\sigma^2_{f_r}(\mathbf{x})$ at $\mathbf{x} \in \mathbf{X}$ is given by:

$$\sigma^2_{f_r}(\mathbf{x}) = \sum_{i=1}^{I} \sigma^2_{f_r^i}(\mathbf{x}) 1_{\mathbf{X}^i}(\mathbf{x}). \tag{3.35}$$

Taking the expectation of this quantity with respect to the input probability density $p(\mathbf{x})$ and averaging over $r = 1, \dots, M$, we obtain

$$\sigma^2_{\mathbf{f},p} := \int_{\mathbf{X}} \sigma^2_{\mathbf{f}}(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} := \int_{\mathbf{X}} \frac{1}{M} \sum_{r=1}^{M} \sigma^2_{f_r}(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}. \tag{3.36}$$

This quantity is a measure of our uncertainty about our prediction over the whole domain $\mathbf{X}$. Notice that, in $\sigma^2_{\mathbf{f},p}$, the uncertainty of the model at $\mathbf{x}$ is weighted by its probability of occurrence $p(\mathbf{x})$. Intuitively speaking, we are willing to accept a somewhat less accurate surrogate in regions of the space occurring with lower probability. Using Equation 3.35, it is straightforward to see that:

$$\sigma^2_{\mathbf{f},p} = \sum_{i=1}^{I} \sigma^2_{\mathbf{f},p^i} P(\mathbf{X}^i), \tag{3.37}$$

where

$$\sigma^2_{\mathbf{f},p^i} := \int_{\mathbf{X}^i} \sigma^2_{\mathbf{f}}(\mathbf{x}) p^i(\mathbf{x}) d\mathbf{x} := \int_{\mathbf{X}} \frac{1}{M} \sum_{r=1}^{M} \sigma^2_{f_r}(\mathbf{x}) p^i(\mathbf{x}) d\mathbf{x},$$

is the uncertainty of our prediction over the element $\mathbf{X}^i$. We refine the element $\mathbf{X}^i$, if the contribution to the global uncertainty coming from it is greater than a certain threshold $\delta > 0$, i.e. we refine $\mathbf{X}^i$ if

$$\sigma^2_{\mathbf{f},p^i} P(\mathbf{X}^i) > \delta. \tag{3.38}$$

As already mentioned, we refine elements by cutting them perpendicular to the most 'important' dimension. At this point, we attempt to give a precise meaning to the concept of 'the most important dimension'. The sensitivity of a real function f with respect to each dimension is captured by the partial derivatives $\frac{\partial f}{\partial x_k}$. Generally speaking, the derivative is significantly different than zero in regions of space where the function varies the most. A quantity that measures the variability of the prediction along the k-th dimension is:

$$V_k^i =:= \left(\frac{1}{M} \sum_{r=1}^{M} \int_{\mathbf{X}^i} \left(\frac{\partial \mu_{f_r}(\mathbf{x})}{\partial x_k} \right)^2 p^i(\mathbf{x}) d\mathbf{x} \right)^{1/2}.$$

V_k^i can be thought as the weighted norm of the sensitivity functions for each output (the weighted been specified by the conditional probability density $p^i(\mathbf{x})$). Furthermore, let us introduce the probability P_k^i that the k-th dimension x_k of a random input point $\mathbf{x} \in \mathbf{X}$ falls inside $\mathbf{X}^i$:

$$P_k^i := \int_{a_k^i}^{b_k^i} \left(\int_{\times_{\ell \neq k}[a_\ell^i, b_\ell^i]} p(\mathbf{x}) dx_1 \ldots, dx_{k-1} dx_{k+1} \ldots dx_K \right) dx_k = \int_{a_k^i}^{b_k^i} p_k(x_k) dx_k, \quad (3.39)$$

where the last equality is a consequence of the independence assumption. We define the *importance* I_k^i of the dimension k of the element $\mathbf{X}^i$ to be

$$I_k^i = V_k^i P_k^i. \quad (3.40)$$

Intuitively, the importance of a particular dimension is proportional to the variability observed along that dimension and to the probability mass along that dimension trapped within the stochastic element. Thus, if $\mathbf{X}^i$ needs refinement (i.e. satisfies Equation 3.38), we cut it perpendicular to the most important dimension k^*, given by

$$k^* = \arg\max_k I_k^i. \quad (3.41)$$

In order to have two new elements with the same probabilities, the splitting point is given by the median of the marginal conditional distribution of $\mathbf{X}^i$ along dimension k given in Equation 3.30.

3.2.8 A complete view at the framework

In this final section, we put together the building blocks of our scheme and discuss the algorithmic details and parallelization strategies. The basic input required is the number of observations per element N and the tolerance $\delta > 0$, used for the refinement criterion (Equation 3.38). Algorithm 3 provides a serial implementation of the scheme.

Algorithm 3: The complete surrogate building framework.

$\mathcal{U} \leftarrow \{(\mathbf{X}, \emptyset, \emptyset)\}$.

$C \leftarrow \emptyset$.

while $\mathcal{U} \neq \emptyset$ **do**

Remove $(\mathbf{X}^i, \mathcal{D}^i, \mathcal{M}^i)$ from $\mathcal{U}$.

if $\mathcal{M}_i = \emptyset$ **then**

Observe N random points drawn from $p^i(\mathbf{x})$ Equation 3.7.

else

Observe $N - |\mathcal{D}^i|$ random points drawn from $p^i(\mathbf{x})$ and add them to $\mathcal{D}^i$.

end if

Fit $\mathcal{M}^i$ using only the data in $\mathcal{D}^i$ (Sections 3.2.1 and 3.2.2).

if Refinement criterion of Equation 3.38 is satisfied for δ **then**

Split $\mathbf{X}^i$ in $\mathbf{X}^{i,1}$ and $\mathbf{X}^{i,2}$ according to Equation 3.41.

Let $\mathcal{D}^{i,1}$ and $\mathcal{D}^{i,2}$ be the set observations in $\mathcal{D}^i$ whose inputs live in $\mathbf{X}^{i,1}$ and $\mathbf{X}^{i,2}$, respectively.

$\mathcal{U} \leftarrow \mathcal{U} \cup \{(\mathbf{X}^{i,1}, \mathcal{D}^{i,1}, \mathcal{M}^i), (\mathbf{X}^{i,2}, \mathcal{D}^{i,2}, \mathcal{M}^i)\}$.

else

$C \leftarrow C \cup \{(\mathbf{X}^i, \mathcal{D}^i, \mathcal{M}^i)\}$.

end if

end while

The scheme works in one element cycles that comprise of collecting observations (random samples from the conditional distribution $p^i(\mathbf{x})$ of the element), fitting (Sections 3.2.1 and 3.2.2) and adapting (Section 3.2.7). Let us denote with $\mathbf{X}^i$ a stochastic element, $\mathcal{D}^i$ the observations made on $\mathbf{X}^i$ and $\mathcal{M}^i$ the MRVM fitted over $\mathbf{X}^i$ using $\mathcal{D}^i$. Let C be the set of triplets $\mathbf{X}^i, \mathcal{D}^i, \mathcal{M}^i$ for which the refinement criterion Equation 3.38 is not satisfied. We will refer to $\mathcal{C}$ as the set of *completed triplets*. The rest of the triplets are put in $\mathcal{U}$, called the set of *uncompleted triplets*. With $|\mathcal{D}^i|$ we denote the number of observations inside $\mathcal{D}^i$.

Parallelization of Algorithm 3 is relatively easy. Each node p, has its own set of completed $\mathcal{C}_p$ and uncompleted $\mathcal{U}_p$ elements. Initially the root node $p = 0$ starts as in Algorithm 3 and the rest with $\mathcal{U}_p = \emptyset, \mathcal{C}_p = \emptyset, p \neq 0$. Then, everything proceeds as in Algorithm 3 with load re-balancing at the end of each outer iteration (uncompleted elements are sent to processors with $\mathcal{U}_p = \emptyset$).

3.3 Numerical examples

All examples are run on massively parallel computers at the National Energy Research Scientific Computing Center (NERSCC). The parallelization strategy is straightforward: each processor is assigned to work with a single element. The communication burden between the processes is minimal. Our implementation utilizes extensively the Trilinos library [48] as well as GSL [30].

The ultimate goal of the numerical examples is to demonstrate that the method can:

1. Learn non-stationary surfaces.
2. Deal with discontinuities.
3. Identify localized features of the response.
4. Reduce sampling frequency on unimportant input dimensions.

In Section 3.3.1 we apply our method to the Krainchnan-Orszag ODE system with random initial conditions. Section 3.3.2 examines the classical stochastic elliptic problem. Finally, in Section 3.3.3, we solve a stochastic flow through porous media problem. In all problems, the underlying input probability distribution $p(\mathbf{x})$ is explicitly stated. All tasks start with a single element (the input domain itself) and N random samples drawn from the input distribution. N, is also the maximum number of samples taken within an element and is different for each example. From that point, the algorithm proceeds until a pre-specified tolerance $\delta > 0$ is reached. The refinement criterion is given by Equation 3.38. The only parameters of the method are N and δ. RVM-SE stands for RVM using Square Exponential kernels and RVM-GPC stands for RVM using optimal orthogonal polynomials. For RVM-GPC, all orthogonal polynomials up to a given degree P are constructed on the fly for each element.

3.3.1 Krainchnan-Orszag three-mode problem

Consider the system of ordinary differential equations [91]

$$\begin{aligned}\frac{dy_1}{dt} &= y_1 y_3,\\ \frac{dy_2}{dt} &= -y_2 y_3,\\ \frac{dy_3}{dt} &= -y_1^2 + y_2^2,\end{aligned}$$

subject to random initial conditions at $t = 0$. This dynamical system is particularly interesting because the response has a discontinuity at the planes $y_1(0) = 0,\ y_2(0) = 0$. We solve the system for the time interval $[0, 10]$ and record the response at time step intervals of $\Delta t = 0.01$. This results in a total of $M = 300$ outputs (100 for each of the three dimensions of the response). We will consider two different cases of increasing difficulty with two and three input dimensions and various input distributions. The results we obtain will be compared to MC estimates with 10^6 samples. Let the MC mean and variance be $m_{r,\mathrm{MC}}$ and $v_{r,\mathrm{MC}}$, respectively, $r = 1, \ldots, 300$. The error of the statistics will be evaluated using the (normalized) L_2 *norm of the error in variance* defined by:

$$\mathrm{E}_{L_2} = \frac{1}{M} \sum_{r=1}^{M} \left(v_{r,\mathrm{MC}} - \hat{v}_r\right)^2, \tag{3.42}$$

where $\hat{v}_r$ is given by Equation 3.32.

We start by considering the two-dimensional problem (KO-2) with stochastic initial conditions defined by:

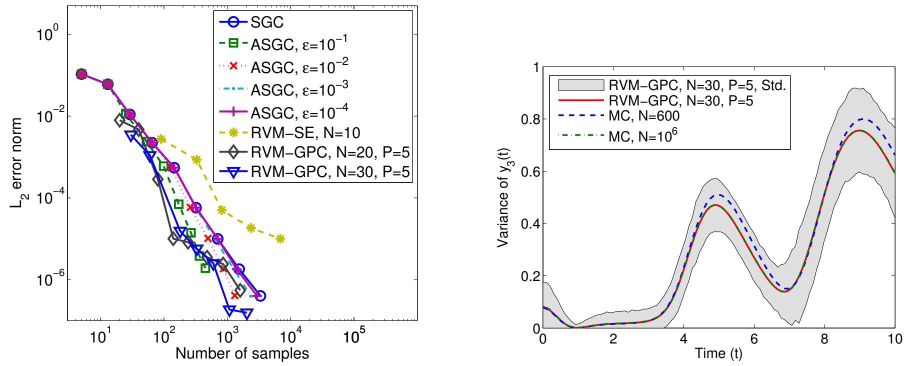
$$y_1(0) = 1,\ y_2(0) = 0.1(2x_1 - 1),\ y_3(0) = 2x_2 - 1,$$

where $x_k, k = 1, 2$ are random variables. We will examine two cases of input distributions, namely:

1. Uniform input: $\mathbf{X} = [0, 1]^2$ and

$$p_k(x_k) = 1,\ k = 1, 2.$$

   Subfig. (a) of Figure 3.1 shows the evolution of the L_2 norm of the error in variance for RVM-SE ($N = 10$) as well as RVM-GPC ($N = 20$) and RMV-GPC ($N = 30$) for maximum polynomial degree $P = 5$ as a function of the number of calls to the deterministic solver. For comparison, we include in the same plot the performance of SGC and ASGC. The $\epsilon > 0$ of ASGC is a parameter specifying the sensitivity of the adaptation criterion (see [59]), that is collocation points with surpluses larger than ϵ spawn new points



(a) L_2 error

(b) Variance Samples

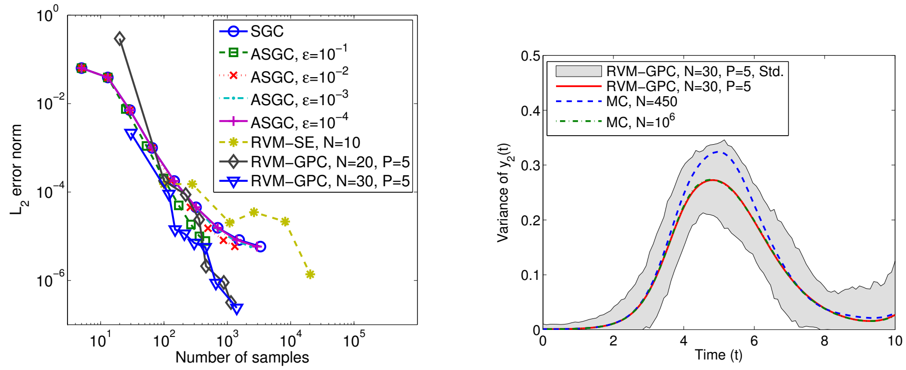
Figure 3.1: KO-2 (Uniform input): (a) L_2 error norm in variance for RVM-SE ($N = 10$), RVM-GPC ($N = 20$) and RVM-GPC ($N = 30$) with maximum polynomial degree $P = 5$, SGC and ASGC with $\epsilon = 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}$. (b) The predictive variance of $y_3(t)$ for RVM-GPC ($N = 30$) with maximum polynomial degree $P = 5$ for a tolerance of $\delta = 10^{-6}$ (red solid line). The total number of observations used by RVM-GPC for this case is 600. The gray area denotes plus/minus one standard deviation of predicted variance estimated from 100 samples of the weights. For comparison, we have included an MC estimator using 600 (dashed blue line) and one using 10^6 observations (dot–dashed green line).

around them. Therefore, for $\epsilon = 0$, one obtains the normal SGC method and as ϵ is increased, less and less collocation points are taken into consideration. The data for the sparse grid based methods are collected as follows: (1) A maximum collocation level is specified (here it is 7); (2) An ϵ is specified ($\epsilon = 0$ for SGC). (3) We add collocation points until the maximum level has been reached or there are no more collocation points with surpluses greater than ϵ. (4) Each time an interpolation level is crossed, we use the number of samples gathered and measure the L_2 error in variance. We observe that RVM-SE is the slowest with performance an order of magnitude worse than SGC. However, RVM-GPC seems to perform at least as well as the fastest ASGC run ($\epsilon = 10^{-1}$). We believe that the poor performance of RVM-SE is due to the poor choice of the kernels and that a scheme that would optimize the evidence also with respect to the length scales on each element would perform better. Subfig. (b) of Figure 3.1 shows the predicted variance of $y_3(t)$ for RVM-GPC ($N = 30$ with maximum polynomial degree $P = 5$ and a tolerance of $\delta = 10^{-6}$), along with plus/minus one standard deviation estimated from 100 samples as described in Section 3.2.6. The number of observations gathered for that particular case is 600 and for comparison we have included the results of a MC estimate using the same number of samples as well as the reference MC estimator based on 10^6 samples. It is apparent that the RVM estimate is much more accurate than the MC result using the same number of samples. However, the error bars are clearly larger than necessary. We believe that this unwanted uncertainty is induced by the underlying assumption of independence of the various output dimensions. The same error bars for the mean are significantly smaller and we have numerically observed that they shrink as the desired tolerance δ decreases, albeit always remaining much wider than the true error.

2. Beta input: $\mathbf{X} = [0, 1]^2$ and

$$p_k(x_k) = \frac{\Gamma(\alpha+\beta)}{2\Gamma(\alpha)\Gamma(\beta)} x_k^{\alpha-1} (1 - x_k)^{\beta-1}, \ k = 1, 2,$$

where $\Gamma(z)$ is the gamma function. We use $\alpha = 2$ and $\beta = 5$. Subfig. (a) of Fig. 3.2 shows the evolution of the L_2 norm of the error in variance for RVM-SE ($N = 10$) as well as RVM-GPC ($N = 20$) and RMV-GPC ($N = 30$) for maximum polynomial degree $P = 5$ as a function of the number of calls to the deterministic solver. In this convergence test, we see again that RVM-SE ($N = 10$) has the worst performance. Notice that it furthermore exhibits some instabilities in the sense that the error is slightly increasing

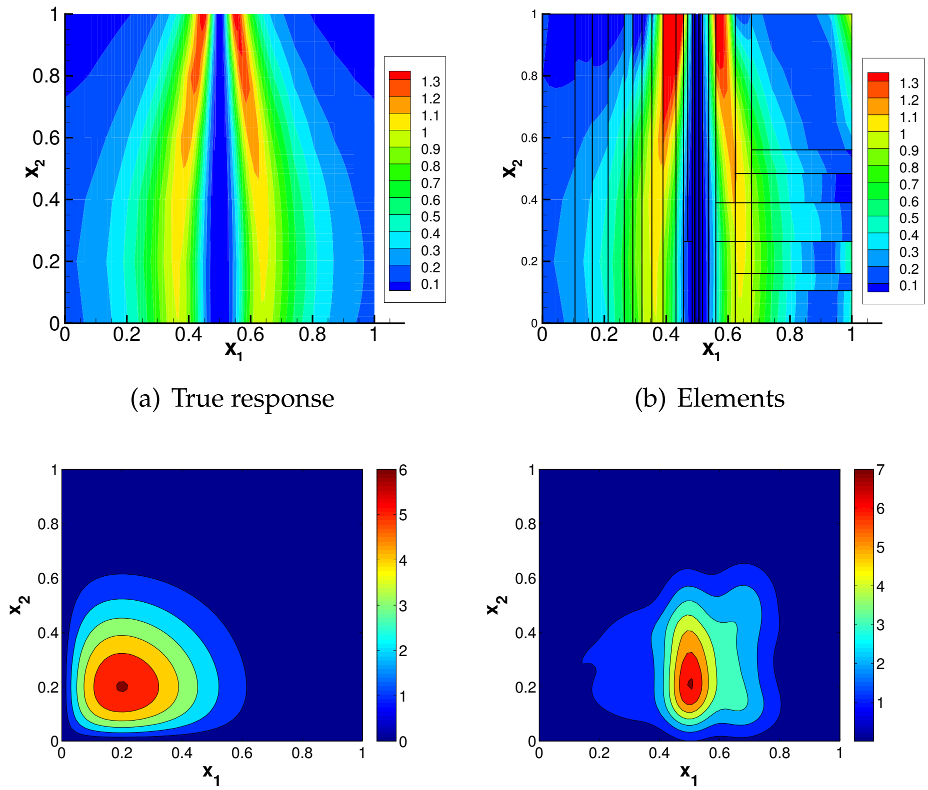


(a) L_2 error

(b) Variance Samples

Figure 3.2: KO-2 (Beta input): (a) L_2 error norm in variance for RVM-SE ($N = 10$), RVM-GPC ($N = 20$) and RVM-GPC ($N = 30$) with maximum polynomial degree $P = 5$, SGC and ASGC with $\epsilon = 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}$. (b) Predictive variance of $y_2(t)$ for RVM-GPC ($N = 30$) with maximum polynomial degree $P = 5$ for a tolerance of $\delta = 10^{-6}$ (red solid line). The total number of observations used by RVM-GPC for this case is 450. The gray area denotes plus/minus one standard deviation of predicted variance estimated from 100 samples of the weights. For comparison, we have included an MC estimator using 450 (dashed blue line) and one using 10^6 observations (dot–dashed green line).

as the number of samples increases as we pass from tolerance $\delta = 10^{-4}$ to $\delta = 10^{-5}$. This is a numerical artifact created by the combination of a poor choice of the kernels as well as N. On the other hand, RVM-GPC converges quite fast with the ($N = 30$) case clearly outperforming ASGC. Subfig. (b) of Fig. 3.2 shows the predicted variance of $y_2(t)$ for RVM-GPC ($N = 30$ with maximum polynomial degree $P = 5$ and a tolerance of $\delta = 10^{-6}$), along with plus/minus one standard deviation estimated from 100 samples as described in Section 3.2.6. The number of observations gathered for that particular case is 450 and for comparison we have included the results of a MC estimate using the same number of samples as well as the reference MC estimator based on 10^6 samples. Again, the RVM estimate is significantly more accurate than the MC result using the same number of samples with the error bars, however, clearly overestimating the true error. For the same RVM-GPC case, Figure 3.3 depicts the stochastic elements (Subfig. (b)) and a kernel density estimator of the probability density of the observed input (Subfig. (d)). For a complete comparison, Subfig. (b) also includes a contour of the prediction at $y_2(t = 10)$ while Subfig. (a) and (c) show the true response of the same output variable and the original input probability density, respectively. Notice how the input probability density affects the decomposition of the stochastic space and -as a result- the selection of the observed samples. The discontinuity is partly resolved. Parts of the response which reside in low probability regions do not have to be fully resolved for an accurate calculation of the statistics.



(a) True response (b) Elements

(c) Input PDF (d) Observed input

Figure 3.3: KO-2 (Beta input): (a) True response $y_2(t = 10)$ as a function of the initial conditions. (b) The discovered elements for RVM-GPC ($N = 30$) for maximum polynomial degree $P = 5$ at a tolerance of $\delta = 10^{-6}$ along with the mean prediction. (c) The original input probability density. (d) A kernel density estimator of the actually observed input points.

Remark 6 *Time dependent discontinuity. In the examples presented above the discontinuity occurs at the same points for all time instants. Since our scheme decomposes the stochastic space based on information coming from all time instants, it will probably lead to very fine decompositions in case the discontinuity moves significantly as a function of time. If one wishes to capture such a situation, we suggest she adds time as one more variable of the surrogate and perform the decomposition in the extended stochastic-time domain. However, this case goes beyond the scope of the current work.*

3.3.2 Elliptic Problem

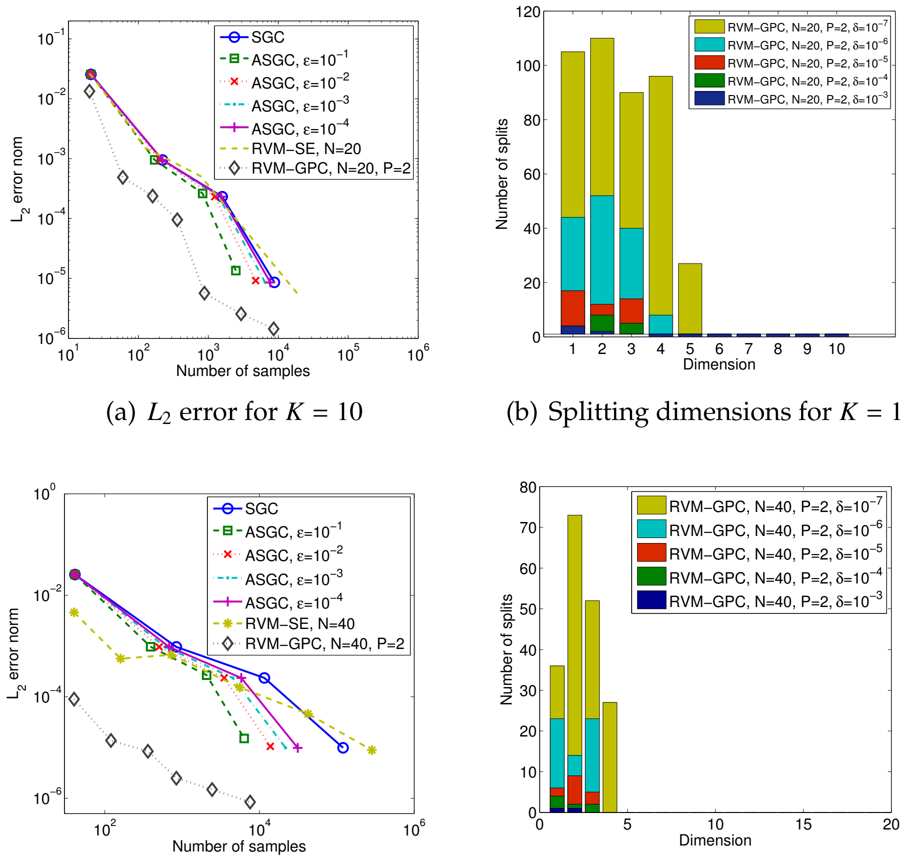
In this section, we consider a simple stochastic elliptic problem [70]. Consider the stochastic partial differential equation (SPDE)

$$\begin{aligned} -\nabla \cdot (a_K(\boldsymbol{\omega}, \cdot)\nabla u(\boldsymbol{\omega}, \cdot)) &= f(\cdot), \quad \text{in } D, \\ u(\boldsymbol{\omega}, \cdot) = 0, \quad \text{on } \partial D, \end{aligned}$$

where the physical domain is $D = [0,1]^2$. In order to avoid confusion with the physical dimension $\mathbf{x}$, we have chosen to denote the random variables with $\boldsymbol{\omega}$ instead of $\mathbf{x}$. We choose a smooth *deterministic* load $f(x,y) = 100\cos(x)\sin(y)$, and work with homogeneous boundary conditions. The deterministic problem is solved with the finite element method using 400 (20×20 grid) bilinear quadrilateral elements. The random diffusion coefficient $a_K(\boldsymbol{\omega}, x)$ is constructed to have a one-dimensional dependence $\log(a_K(\boldsymbol{\omega}, x, y) - 0.5) = 1 + \omega_1 \left(\frac{\sqrt{\pi}L}{2}\right)^{1/2} + \sum_{k=2}^{K} \xi_k \phi_k(x)\omega_k$, where $\xi_k := \left(\sqrt{\pi}L\right)^{1/2} \exp\left(\frac{-\left(\lfloor \frac{k}{2} \rfloor \pi L\right)^2}{8}\right)$, for $k \geq 2$, and

$$\phi_k(x) := \begin{cases} \sin\left(\frac{\lfloor \frac{k}{2} \rfloor \pi x}{L_p}\right) & \text{, if } k \text{ is even,} \\ \cos\left(\frac{\lfloor \frac{k}{2} \rfloor \pi x}{L_p}\right) & \text{, if } k \text{ is odd.} \end{cases}$$

We choose the $\omega_k, k = 1, \ldots, K$ to be independent identically distributed random variables $\omega_k \sim U([-\sqrt{3}, \sqrt{3}])$. Hence, the stochastic input space is $\mathbf{\Omega} = [-\sqrt{3}, \sqrt{3}]^K$. Finally, we set $L_p = \max\{1, 2L_c\}$ and $L = \frac{L_c}{L_p}$, where L_c is called the *correlation length*.



(a) L_2 error for $K = 10$

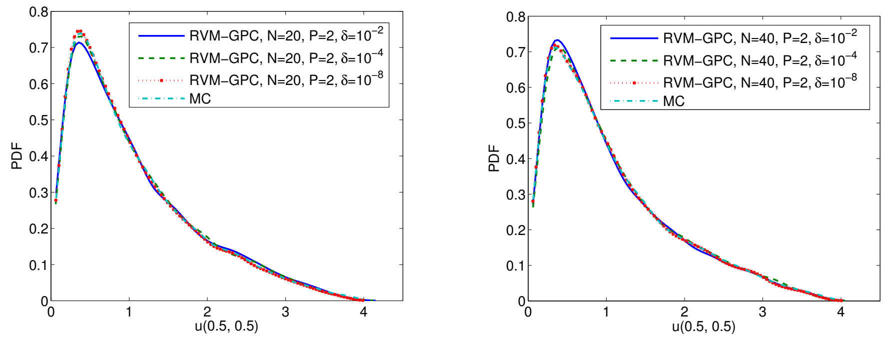
(b) Splitting dimensions for $K = 10$

(c) L_2 error for $K = 20$

(d) Splitting dimensions for $K = 20$

Figure 3.4: The left column shows L_2 error norm in variance for $K = 10$ (a) and $K = 20$ (c) for RVM-SE, RVM-GPC with maximum polynomial degree $P = 2$, SGC and ASGC with $\epsilon = 10^{-1}, 10^{-2}, 10^{3-}$ and 10^{-4}. The right columns shows the number of splits per dimension performed by RVM-GPC for $K = 10$ (b) and $K = 20$ (d).

In this study, we set the correlation length to $L_c = 0.6$ and test the convergence of our method for $K = 10$ and 20 input dimensions. The $K = 10$ and 20 cases are solved using RVM-SE and RVM-GPC up to maximum polynomial degree $P = 2$ for $N = 20$ and 40, respectively, up to a tolerance of 10^{-8}. Figs. 3.4 (a) and (c) compare the convergence of our method to ASGC for $K = 10$ and $K = 20$, respectively. The reference variance was calculated using a MC estimator with 10^6 samples. We observe, that RVM-SE performs similar to ASGC while RVM-GPC seems to converge much faster particularly for the high-dimensional case. In Subfigures (b) and (d) of the same figure we show the number of splits per dimension for various tolerances δ of the RVM-GPC case for $K = 10$ and $K = 20$ input dimensions, respectively. It is clearly seen that the method identifies only the first few dimensions as important, while completely ignoring the rest. Finally, we calculate the predictive PDFs for selected outputs for RVM-GPC. The results are computed as follows: At a given tolerance, we draw $10,000$ random input samples and propagate them through the surrogate. Based on these samples, we fit a kernel density estimator (see [84]) and compare it to the kernel density estimator obtained by using the full deterministic solver instead of the surrogate (this is referred to as "MC" on figure legends). Because of the limited amount of samples used for the kernel density estimation, a small variation on the outcome is expected. Subfigures (a), (b) of Fig. 3.5 depict the PDFs of $u(0.5, 0.5)$ for $K = 10$ and $K = 20$ input dimensions, respectively. As a general rule, a smaller tolerance is required to capture a qualitatively similar PDF than it is to capture the correct variance. At this point, we must mention that the PDFs produced by RVM-SE are not quite as satisfactory. This is due to the known limitations of the SE kernels: (1) It is very difficult to identify the right length scales; (2) In high dimensions, the SE kernel is not a good similarity measure between input points [29].



(a) PDF of $u(0.5, 0.5)$ for $K = 10$ (b) PDF of $u(0.5, 0.5)$ for $K = 20$

Figure 3.5: Elliptic Problem: Comparison of the predicted PDF of $u(0.5, 0.5)$ using various RVM-GPC surrogates with the MC predictions. (a) $K = 10$ input dimensions. (b) $K = 20$ input dimensions.

3.3.3 Flow through porous media

Consider a bounded two dimensional spatial domain $D \subset \mathbb{R}^2$ with smooth boundary ∂D. The governing equations for immiscible and incompressible two-phase flow in porous media consists of an elliptic equation for fluid pressure and a transport equation for the movement of fluid phases. For simplicity, we neglect the effects from gravity, capillary forces and assume that the porosity is a constant. The two phases will be referred to as water and oil, denoted by w and o, respectively. The total Darcy velocity $\mathbf{u}$ and the pressure p satisfy [53]:

$$\nabla \cdot \mathbf{u} = 0, \ \mathbf{u} = -\alpha(\mathbf{x}, \boldsymbol{\omega})\lambda_t \nabla p, \forall x \in D, \tag{3.43}$$

with the following boundary conditions:

$$p = \bar{p}, \text{ on } \partial D_p, \text{ and } \mathbf{u} \cdot \mathbf{n} = \bar{\mathbf{u}} \text{ on } \partial D_u. \tag{3.44}$$

The total velocity $u = u_o + u_w$ is the sum of the velocities of oil $\mathbf{u}_o$ and water $\mathbf{u}_w$. The random permeability $a(\mathbf{x}, \cdot)$ is assumed to be diagonal and uniformly positive definite. In addition, we will assume that $a(\mathbf{x}, \cdot)$ is a stochastic scalar function. The total mobility is given by $\lambda_t = \lambda_w + \lambda_o$, where λ_i models the reduced mobility of phase i due to the presence of the other phase. Without loss of generality, we assume that the boundary conditions are deterministic and that $\bar{\mathbf{u}} \cdot \mathbf{n} = 0$ on ∂D_u, where $\mathbf{n}$ is the unit normal of ∂D_u. Furthermore, we use the unit mobility displacement model, i.e. $\lambda_w = S, \lambda_o = 1 - S$ and hence $\lambda_t = 1$, where S is the water saturation. Under these assumptions, the water saturation equation is given by

$$\frac{\partial S(\mathbf{x}, t, \omega)}{\partial t} + \mathbf{u} \cdot \nabla S(\mathbf{x}, t, \omega) = 0, \ \forall x \in D, t \in [0, T]. \tag{3.45}$$

Geostatistical models often suggest that the permeability field is a weakly stationary second-order random field such that the mean log-permeability $G(\mathbf{x}, \cdot) = \ln a(\mathbf{x}, \cdot)$ is constant and its covariance function only depends on the relative distance of two points [15]:

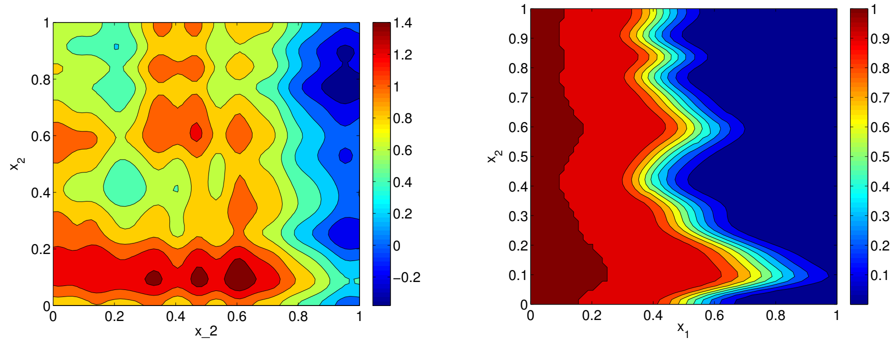
$$\mathrm{Cov}(\mathbf{x}, \mathbf{y}) = \exp\left\{-\frac{|x_1 - y_1|}{L_1} - \frac{|x_2 - y_2|}{L_2}\right\},$$

where $L_i, i = 1, 2$ are the correlation lengths. Employing the finite-dimensional noise assumption [95] and the Karhunen-Loève (KL) expansion [33], we approximate the log permeability field via a finite-dimensional representation:

$$G(\mathbf{x}, \omega) = \sum_{k=1}^{K} \sqrt{\lambda_k}\phi_k(\mathbf{x})\omega_k, \tag{3.46}$$

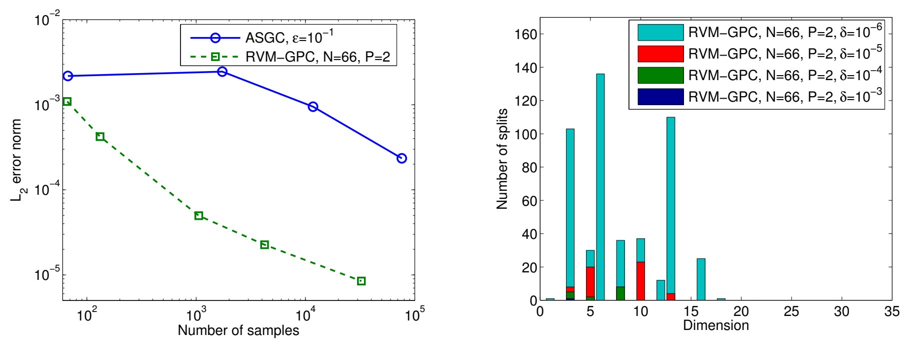
where ω_k are uncorrelated random variables, while $\phi_k(\mathbf{x})$ and λ_k are the eigenfunctions and eigenvalues of the covariance function, respectively, which are analytically available [54]. According to the KL expansion, $\omega_k \sim \mathcal{N}(0, 1)$. However, we may assume that $\omega_k \sim U(-1, 1)$ without losing the main features of the output uncertainty [55]. Note that such a restriction is not necessary for our approach (we could as well use the Gaussian distribution), but we use it so that we can solve the same problem using ASGC. The deterministic problem for the velocity defined in Equation 3.44 is solved with a mixed finite element method [31, 60] on the spatial domain $D = [0, 1]^2$ utilizing a 64×64 fine grid. The boundary conditions are set by fixing the pressure to 1 on the left boundary and 0 one the right boundary and using $\bar{\mathbf{u}} \cdot \mathbf{n} = 0$ on the top and the bottom. Given the velocity field $\mathbf{u}$, we solve the saturation equation (Equation 3.45) following a discontinuous Galerkin approach with piecewise constant elements [26] coupled with a simple Euler scheme. The initial saturation is set to zero, while it is kept fixed to 1 on the left side of the boundary. Our C++ solver is built upon FEniCS [57]. The response is taken to be the value of the saturation S at time $t = 0.5$ PVI (see [60] for the definition of PVI) on each finite element node, that is the problem has $M = 64 \times 64 = 4096$ output dimensions.

The stochastic problem is solved for correlation lengths $L_i = 1$ $(i = 1, 2)$, requiring $K = 33$ input dimensions to account for 95% of the field's energy. In Figure 3.6, we plot a sample permeability field along with the corresponding saturation. In this case, we use only RVM-GPC with $N = 66$ samples per element and polynomials of maximum degree $P = 2$. In Subfig. (a) of Figure 3.7 we plot the L_2 error in variance as a function of the number of observations and compare it to ASGC with $\epsilon = 10^{-1}$. The reference variance was calculated using a MC estimator based on 860,160 samples. It is clear that RVM-GPC outperforms ASGC by at least two orders of magnitude. Subfig. (b) of the same figure shows the number of splits per dimension performed by RVM-GPC. As in the results of the elliptic problem, we observe that RVM-GPC puts more weight on the first few dimensions while ignoring the rest. Finally, Figure 3.8 compares the predicted statistics against the corresponding MC estimates.



(a) Sample log permeability (b) Sample saturation

Figure 3.6: Saturation field: (a) A random sample of the permeability. (b) The corresponding saturation at $t = 0.5$ PVI.

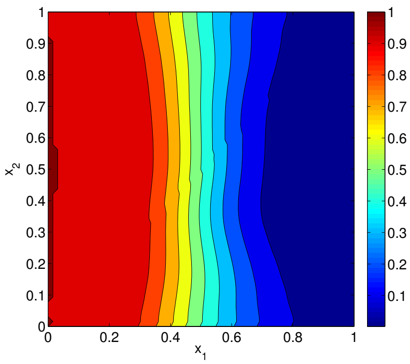


(a) L_2 error (b) Splitting dimensions

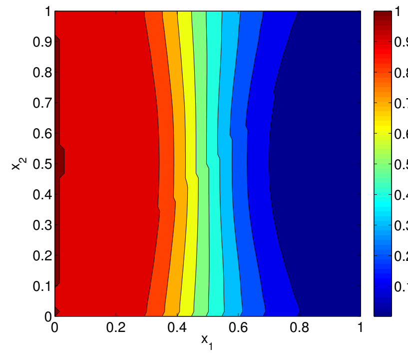
Figure 3.7: Saturation field: (a) The L_2 error in variance for RVM-GPC ($N = 66$) with maximum polynomial degree $P = 2$ and ASGC. (b) The number of splits per dimension performed by RVM for a tolerance of $\delta = 10^{-6}$.

3.4 Conclusions

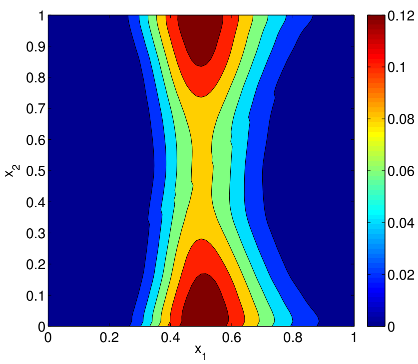
We have developed a UQ framework based on local Bayesian regression models. The stochastic space is adaptively decomposed in fine elements based solely on information conveyed by the local regressors. The adaptivity criteria developed are applicable to arbitrary input probability distributions and any local Bayesian regression model. The Bayesian regression we used was based on an



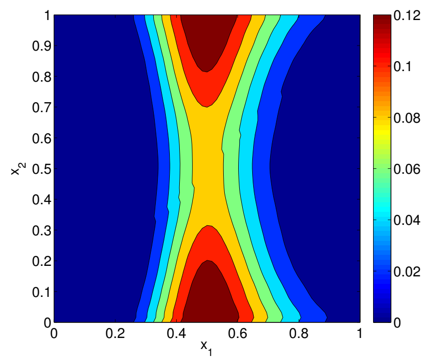
(a) RVM-GPC mean saturation



(b) MC mean saturation



(c) RVM-GPC variance of saturation



(d) MC variance of saturation

Figure 3.8: Saturation field: The left column shows the statistics predicted by RVM-GPC ($N = 33$ with maximum polynomial degree $P = 2$ at a tolerance $\delta = 10^{-6}$) while the right column shows the corresponding results using $860,160$ MC samples. The top row ((a) and (b)) shows the mean and the bottom row ((c) and (d)), the variance of the saturation.

extension of the RVM model that accounts for the multiple dimensions of the output (MRVM). A fast algorithm was developed to train MRVM on a given data set that does not require matrix inversions. The Bayesian nature of the scheme allowed us to sample from the predictive distribution of the desired statistics, thereby quantifying the epistemic uncertainty introduced by replacing the deterministic solver with a surrogate. The scheme was demonstrated through various numerical examples and its ability to capture discontinuities was verified. In high-dimensional input settings ($K > 20$), the optimal orthogonal polynomial basis performed much better especially in correctly identifying the PDFs of the various outputs. In the future, we plan to investigate the way the choice of N affects the results and identify ways for an automatic optimal choice for it.

CHAPTER 4

MULTI-OUTPUT GAUSSIAN PROCESS REGRESSION WITH SPATIO-TEMPORAL CORRELATIONS

4.1 Introduction

In this chapter, we are mainly concerned with responses that are multi-output functions of space and/or time. In Chapter 2, we saw that one derive analytic estimates of the epistemic uncertainty induced by the limited number of observations and noticed that these approximate bounds are significantly wide. Here, instead of relying on analytic estimates we develop a numerical schemes that is able to sample from the posterior of the statistics of interest. Even though we are making use of active learning ideas in a completely different context (see Section 4.2.3), we will be assuming that the observations are simply given to us. Our first goal is the construction of a multi-output Gaussian process model that explicitly treats space and time (Section 4.2.1). This model, in its full generality is extremely computationally demanding. In Section 4.2.2, we carefully develop the so-called separable model, which allows us to express the inference and prediction tasks using Kronecker products of small matrices. This, in turn, results in highly efficient computations. Finally, in Section 4.2.3, we apply our scheme to uncertainty quantification tasks. Contrary to other approaches, we recognize the fact that the predictive distribution of the Gaussian process conditional on the observations, actually defines a probability measure on the function space of possible surrogates. The weight of this measures corresponds to the epistemic uncertainty induced by the limited data. Extending on ideas of [71], we develop a procedure that allows us to approximately sample this probability space. Each sample, is a kernel approximation of a candidate surrogate for the code. In the same section, we show how we can semi-analytically compute all the statistics of the candidate surrogate up to second order. Higher order statistics or even probability densities may be calculated quite effortlessly via a Monte Carlo procedure. By repeatedly sampling the posterior surrogate space, we are able to provide error bars for practically anything that depends on it. In Section 4.3.1, we apply our scheme to a stochastic ordinary differential equation with three distinct outputs and two random variables. The purpose of this example, is to demonstrate the validity of our approach in a simple problem. In Section 4.3.2, we consider the problem of flow through random porous media. There, we model the velocity field and the pressure as a function of space and 50 random variables. In this more challenging problem, we clearly see the advantages of a fully Bayesian approach to uncertainty quantification. Namely, the ability to say something about the statistics of a 50 dimensional stochastic problem with as little as 24 observations is intriguing. Finally, we conclude by noting the limitations of the approach and discuss the many possibilities for extension.

4.2 Methodology

We are interested in modeling computer codes returning a multi-output response $\mathbf{y} \in \mathbb{R}^q$, where $q > 0$ is the number of distinct outputs, given an input $\boldsymbol{\xi} \in \mathcal{X}_{\xi} \subset \mathbb{R}^{k_{\xi}}$, defined over a spatial domain $\mathcal{X}_s \subset \mathbb{R}^{k_s}$ and/or a time interval $\mathcal{X}_t = [0, T]$, where k_{ξ} is the number of inputs to the code, k_s the spatial dimension (either 1, 2 or 3) and $T > 0$ is the time horizon. Even though for a given input $\boldsymbol{\xi}$ the code reports the response simultaneously at various spatial and time locations, we will be modeling it as a function $\mathbf{f} : \mathcal{X}_{\xi} \times \mathcal{X}_s \times \mathcal{X}_t \to \mathbb{R}^q$. As an example, you may consider the problem of two-dimensional flow in random porous media. The input variables $\boldsymbol{\xi}$ would represent the permeability field while - for a fixed $\boldsymbol{\xi}$ - $\mathbf{f}(\boldsymbol{\xi}, \mathbf{x}_s, t)$ would give the velocity components as well as the pressure at the spatial location $\mathbf{x}_s$ and time t (here $q = 3$).

4.2.1 Multi-output Gaussian process regression

Throughout this work, we will collectively denote the inputs of $\mathbf{f}(\cdot)$ by:

$$\mathbf{x} = (\boldsymbol{\xi}, \mathbf{x}_s, t),$$

and its input domain by $\mathcal{X} = \mathcal{X}_\xi \times \mathcal{X}_s \times \mathcal{X}_t \subset \mathbb{R}^k$, where $k = k_\xi + k_s + 1$. Following [19], we model $\mathbf{f}(\cdot)$ as a q-dimensional Gaussian process:

$$\mathbf{f}(\cdot)|\mathbf{B}, \boldsymbol{\Sigma}, \boldsymbol{\theta} \sim \mathcal{N}_q\left(\mathbf{m}(\cdot; \mathbf{B}), c(\cdot, \cdot; \boldsymbol{\theta})\boldsymbol{\Sigma}\right), \tag{4.1}$$

conditional on hyper-parameters $\mathbf{B} \in \mathbb{R}^{m\times q}, \boldsymbol{\Sigma} \in \mathbb{R}^{q\times q}$ and $\boldsymbol{\theta}$, with $\boldsymbol{\theta}$ denoting the parameters of the correlation function $c(\cdot, \cdot; \boldsymbol{\theta})$. Notice that Eq. (4.1) essentially means that:

$$\mathbf{E}\left[\mathbf{f}(\mathbf{x})|\mathbf{B}, \boldsymbol{\Sigma}, \boldsymbol{\theta}\right] = \mathbf{m}(\mathbf{x}; \mathbf{B}),$$

and

$$\mathrm{Cov}\left[\mathbf{f}(\mathbf{x}_1), \mathbf{f}(\mathbf{x}_2)|\mathbf{B}, \boldsymbol{\Sigma}, \boldsymbol{\theta}\right] = c(\mathbf{x}_1, \mathbf{x}_2; \boldsymbol{\theta})\boldsymbol{\Sigma}.$$

It is apparent that $\boldsymbol{\Sigma}$ must be a symmetric, positive definite matrix and that it models the linear part of the correlations between the q distinct outputs. The mean is chosen to be a *generalized linear model*:

$$\mathbf{m}(\mathbf{x}; \mathbf{B}) = \mathbf{B}^T\mathbf{h}(\mathbf{x}), \tag{4.2}$$

where $\mathbf{h} : \mathcal{X} \to \mathbb{R}^m, \mathbf{h}(\cdot) = (h_1(\cdot), \dots, h_m(\cdot))$ are regression functions shared by each of the components of $\mathbf{f}(\cdot)$.

Prior distributions We are assuming that a priori the pair $(\mathbf{B}, \mathbf{\Sigma})$ and $\boldsymbol{\theta}$ are independent:

$$\pi(\mathbf{B}, \mathbf{\Sigma}, \boldsymbol{\theta}) = \pi(\mathbf{B}, \mathbf{\Sigma})\pi(\boldsymbol{\theta}).$$

For the moment, we let the prior $\pi(\boldsymbol{\theta})$ of $\boldsymbol{\theta}$ to be undefined. For $\mathbf{B}$ and $\mathbf{\Sigma}$, we choose to use the "non-informative" prior [19]:

$$\pi(\mathbf{B}, \mathbf{\Sigma}) \propto |\mathbf{\Sigma}|^{-\frac{q+1}{2}}. \tag{4.3}$$

Notice, that the prior corresponding to $\mathbf{\Sigma}$ is a pathological case of an Inverse Wishart distribution. The real reason for this choice is that it leads to an analytically tractable model, since -as is shown in what follows- both $\mathbf{B}$ and $\mathbf{\Sigma}$ can be integrated out. Of course, one might choose an alternative prior distribution that reflects her own beliefs at the cost of having to numerically sample these two parameters.

The likelihood of the data Given a data set of n observations with inputs $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n)^T \in \mathbb{R}^{n\times k}$ and outputs $\mathbf{Y} = (\mathbf{f}(\mathbf{x}_1), \dots, \mathbf{f}(\mathbf{x}_n))^T \in \mathbb{R}^{n\times q}$, it follows from Equation 4.1 that the likelihood is given by the matrix-normal (see [24]):

$$\mathbf{Y}|\mathbf{B}, \mathbf{\Sigma}, \boldsymbol{\theta} \sim \mathcal{N}_{n\times q}\left(\mathbf{H}\mathbf{B}, \mathbf{\Sigma}, \mathbf{A}\right), \tag{4.4}$$

where

$$\mathbf{H} = (\mathbf{h}(\mathbf{x}_1), \dots, \mathbf{h}(\mathbf{x}_n))^T \in \mathbb{R}^{n\times m}, \tag{4.5}$$

is the *design matrix* and

$$\mathbf{A} = \left(c(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\theta})\right) \in \mathbb{R}^{n\times n}, \tag{4.6}$$

is the usual *covariance matrix*.

The predictive distribution Using the vectorization operation [63] to cast the matrix-normal distribution to a simple multivariate normal and some trivial identities (for example see Chapter 2.3.1 of [7]), it is easy to show that the predictive distribution is given by:

$$\mathbf{f}(\cdot)|\mathbf{B}, \boldsymbol{\Sigma}, \boldsymbol{\theta}, \mathbf{Y} \sim \mathcal{N}_q\left(\mathbf{m}^*(\cdot, \mathbf{B}), c^*(\cdot, \cdot; \boldsymbol{\theta})\boldsymbol{\Sigma}\right), \tag{4.7}$$

where

$$\begin{aligned}
\mathbf{m}^*(\mathbf{x}; \mathbf{B}) &= \mathbf{B}^T\mathbf{h}(\mathbf{x}) + (\mathbf{Y} - \mathbf{H}\mathbf{B})^T\mathbf{A}^{-1}\mathbf{a}(\mathbf{x}),\\
\mathbf{c}^*(\mathbf{x}_1, \mathbf{x}_2; \boldsymbol{\theta}) &= c(\mathbf{x}_1, \mathbf{x}_2; \boldsymbol{\theta}) - \mathbf{a}^T(\mathbf{x}_1)\mathbf{A}^{-1}\mathbf{a}(\mathbf{x}_2),
\end{aligned}$$

where $\mathbf{a}(\cdot) = (c(\cdot, \mathbf{x}_1; \boldsymbol{\theta}), \dots, c(\cdot, \mathbf{x}_n; \boldsymbol{\theta})) \in \mathbb{R}^n$. If $n > m + q$ (so that all distributions involved are proper), it is possible to integrate out both $\mathbf{B}$ and $\boldsymbol{\Sigma}$ resulting in the predictive distribution of $\mathbf{f}(\cdot)$ conditional only on $\boldsymbol{\theta}$. It is a q-variate Student's process with $n - m$ degrees of freedom (see [19]):

$$\mathbf{f}(\cdot)|\boldsymbol{\theta}, \mathbf{Y} \sim \mathcal{T}_q\left(\mathbf{m}^{**}(\cdot), c^{**}(\cdot, \cdot; \boldsymbol{\theta})\hat{\boldsymbol{\Sigma}}; n - m\right), \tag{4.8}$$

where

$$\begin{aligned}
\mathbf{m}^{**}(\mathbf{x}) &= \hat{\mathbf{B}}^T\mathbf{h}(\mathbf{x}) + (\mathbf{Y} - \mathbf{H}\hat{\mathbf{B}})^T\mathbf{A}^{-1}\mathbf{a}(\mathbf{x}),\\
\mathbf{c}^{**}(\mathbf{x}_1, \mathbf{x}_2; \boldsymbol{\theta}) &= c^*(\mathbf{x}_1, \mathbf{x}_2; \boldsymbol{\theta}) +\\
&\quad \left(\mathbf{h}(\mathbf{x}_1) - \mathbf{H}^T\mathbf{A}^{-1}\mathbf{a}(\mathbf{x}_1)\right)^T\left(\mathbf{H}^T\mathbf{A}^{-1}\mathbf{H}\right)^{-1}\left(\mathbf{h}(\mathbf{x}_2) - \mathbf{H}^T\mathbf{A}^{-1}\mathbf{a}(\mathbf{x}_2)\right),\\
\hat{\mathbf{B}} &= \left(\mathbf{H}^T\mathbf{A}^{-1}\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{A}^{-1}\mathbf{Y},\\
\hat{\boldsymbol{\Sigma}} &= = \frac{1}{n - m}\left(\mathbf{Y} - \mathbf{H}\hat{\mathbf{B}}\right)^T\mathbf{A}^{-1}\left(\mathbf{Y} - \mathbf{H}\hat{\mathbf{B}}\right).
\end{aligned}$$

The posterior distribution of $\boldsymbol{\theta}$ Let us conclude this section by giving the posterior distribution of the hyper-parameters of the correlation function. Using

Bayes theorem to write down the joint posterior for $\mathbf{B}, \mathbf{\Sigma}$ and $\boldsymbol{\theta}$ conditional on $\mathbf{Y}$ (combining Eqs. (4.3) and (4.4)) and integrating out $\mathbf{B}$ and $\mathbf{\Sigma}$, we obtain:

$$p(\boldsymbol{\theta}|\mathbf{Y}) \propto \pi(\boldsymbol{\theta})|\mathbf{A}|^{-\frac{q}{2}}|\mathbf{H}^T\mathbf{A}^{-1}\mathbf{H}|^{-\frac{q}{2}}|\hat{\mathbf{\Sigma}}|^{-\frac{n-m}{2}}. \tag{4.9}$$

4.2.2 The separable model

It is apparent that the above mentioned model becomes computationally intractable quite fast due to the fact that a high-dimensional and dense covariance matrix has to be inverted. An important simplification can be achieved if the spatial and the temporal points at which the output is observed remain fixed independent of the input $\boldsymbol{\xi}$ and if we assume that the correlation function is separable, i.e.:

$$c(\mathbf{x}_1, \mathbf{x}_2; \boldsymbol{\theta}) := c_{\xi}(\boldsymbol{\xi}_1, \boldsymbol{\xi}_2; \boldsymbol{\theta}_{\xi})c_s(\mathbf{x}_{s,1}, \mathbf{x}_{s,2}; \boldsymbol{\theta}_s)c_t(t_1, t_2; \boldsymbol{\theta}_t), \tag{4.10}$$

where $c_{\xi}(\cdot,\cdot;\boldsymbol{\theta}_{\xi}), c_s(\cdot,\cdot;\boldsymbol{\theta}_s)$ and $c_t(\cdot,\cdot;\boldsymbol{\theta}_t)$ are the correlation functions of the parameter space, the spatial domain and the time domain, respectively, and $\boldsymbol{\theta} = (\boldsymbol{\theta}_{\xi}, \boldsymbol{\theta}_s, \boldsymbol{\theta}_t)$. We will now show that under these assumptions, the covariance matrix can be written as the Kronecker product of smaller covariance matrices. Using this observation, it is possible to carry out inference and make predictions without ever forming the full covariance matrix. Finally, we also assume that the hyper-parameters of the various covariance functions are a priori independent:

$$\pi(\boldsymbol{\theta}) = \pi(\boldsymbol{\theta}_{\xi})\pi(\boldsymbol{\theta}_s)\pi(\boldsymbol{\theta}_t). \tag{4.11}$$

Remark 1: Another more general model for the covariance function is the linear model of coregionalization (LMC) [38, 9, 25]. The more general nature of this covariance function does not necessarily make it more attractive for the applications of interest. The introduction of such models is usually associated with higher computational cost which we try to avoid in this paper.

Organizing the inputs Let us consider how the data are collected from a computer code. For a parameter $\boldsymbol{\xi} \in \mathcal{X}_{\xi}$, the computer code returns the (multi-output) response on a given (a priori known) set of n_s spatial points $\mathbf{X}_s = (\mathbf{x}_{s,1}, \dots, \mathbf{x}_{s,n_s})^T \in \mathbb{R}^{n_s \times k_s}$, where $k_s = 1, 2$ or 3 is the spatial dimension ($\mathcal{X}_s \subset \mathbb{R}^{k_s}$), at each one of the n_t timesteps $\mathbf{X}_t = (t_1, \dots, t_{n_t}) \in \mathbb{R}^{n_t \times 1}$. That is, a single choice of the parameter $\boldsymbol{\xi}$ generates a total of $n_s n_t$ training samples. Therefore, the response of the code is a matrix $\mathbf{Y}_{\xi} \in \mathbb{R}^{(n_s n_t) \times q}$, which we call the *output matrix*. The output matrix is assembled as follows:

$$\mathbf{Y}_{\xi} = (\mathbf{y}_{\xi,1}^T \dots \mathbf{y}_{\xi,n_s}^T)^T,$$

where each $\mathbf{y}_{\xi,i} \in \mathbb{R}^{n_t \times q}$ is the response at the spatial point $\mathbf{x}_{s,i}$ at each timestep, that is:

$$\mathbf{y}_{\xi,i} = (\mathbf{y}_{\xi,i,1} \dots \mathbf{y}_{\xi,i,n_t})^T,$$

where $\mathbf{y}_{\xi,i,j} \in \mathbb{R}^q$ is the response at the spatial point $\mathbf{x}_{s,i}$ at time t_j:

$$\mathbf{y}_{\xi,i,j} = (y_{\xi,i,j,1} \dots y_{\xi,i,j,q})^T,$$

where, of course, $y_{\xi,i,j,l}$ is the l-th output of the response at $\mathbf{x}_{s,i}$ at t_j.

Separating the covariance matrices If we take a total of n_ξ samples of the parameters:

$$\mathbf{X}_{\xi} = (\boldsymbol{\xi}_1, \dots, \boldsymbol{\xi}_{n_\xi})^T \in \mathbb{R}^{n_\xi \times k_\xi},$$

where k_ξ is the dimension of the input variables $\boldsymbol{\xi}$ ($\mathcal{X}_\xi \subset \mathbb{R}^{k_\xi}$), we will have a total of

$$n = n_\xi n_s n_t$$

training samples for our model. The covariance matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ can now be written as:

$$\mathbf{A} = \mathbf{A}_\xi \otimes \mathbf{A}_s \otimes \mathbf{A}_t, \tag{4.12}$$

where $\mathbf{A}_\xi \in \mathbb{R}^{n_\xi \times n_\xi}$ is the covariance matrix generated by $\mathbf{X}_\xi$ and $c_\xi(\cdot,\cdot;\boldsymbol{\theta}_\xi)$, $\mathbf{A}_s \in \mathbb{R}^{n_s\times n_s}$ is the covariance matrix generated by $\mathbf{X}_s$ and $c_s(\cdot;\cdot;\boldsymbol{\theta}_s)$ and $\mathbf{A}_t \in \mathbb{R}^{n_t\times n_t}$ is the covariance matrix generated by $\mathbf{X}_t$ and $c_t(\cdot;\cdot;\boldsymbol{\theta}_t)$ and $\otimes$ corresponds to the Kronecker product.

Separating the design matrices Now let us consider the basis functions used in the generalized linear model of Equation 4.2. Suppose, we wish to use m_t basis functions to capture the time dependence of the mean:

$$\mathcal{H}_t = \{h_{t,1}(t), \dots, h_{t,m_t}(t)\}.$$

We choose also m_s basis functions to capture the spatial dependence of the mean:

$$\mathcal{H}_s = \{h_{s,1}(\mathbf{x}_s), \dots, h_{s,m_s}(\mathbf{x}_s)\}.$$

These can be for example the finite element basis of the model or any other suitable functions. Finally, we choose m_ξ basis functions to capture the dependence of the mean on the stochastic parameter:

$$\mathcal{H}_\xi = \{h_{\xi,1}(\boldsymbol{\xi}), \dots, h_{\xi,m_\xi}(\boldsymbol{\xi})\}.$$

For example, in an uncertainty quantification setting these could be a gPC basis as induced by the probability distribution of the $\boldsymbol{\xi}$'s. The global basis functions are formed from the tensor product:

$$\mathcal{H} = \mathcal{H}_{\xi} \otimes \mathcal{H}_s \otimes \mathcal{H}_t.$$

Thus, the total number of basis functions present in the model is:

$$m = m_{\xi} m_s m_t.$$

In order to have a consistent enumeration, we proceed as follows:

$$\begin{aligned}
h_1(\mathbf{x}) &:= h_{\xi,1}(\boldsymbol{\xi}) h_{s,1}(\mathbf{x}_s) h_{t,1}(t),\\
h_2(\mathbf{x}) &:= h_{\xi,1}(\boldsymbol{\xi}) h_{s,1}(\mathbf{x}_s) h_{t,2}(t),\\
&\vdots\\
h_{m_t}(\mathbf{x}) &:= h_{\xi,1}(\boldsymbol{\xi}) h_{s,1}(\mathbf{x}_s) h_{t,m_t}(t),\\
h_{m_t+1}(\mathbf{x}) &:= h_{\xi,1}(\boldsymbol{\xi}) h_{s,2}(\mathbf{x}_s) h_{t,1}(t),\\
&\vdots\\
h_{m_s m_t (i-1) + m_t (j-1) + l} &:= h_{\xi,i}(\boldsymbol{\xi}) h_{s,j}(\mathbf{x}_s) h_{t,l}(t),
\end{aligned}$$

where, at the last line, $i = 1, \dots, m_{\xi}$, $j = 1, \dots, m_s$ and $l = 1, \dots, m_t$. With this enumeration, the design matrix $\mathbf{H}$ defined in Equation 4.5 breaks down as:

$$\mathbf{H} = \mathbf{H}_{\xi} \otimes \mathbf{H}_s \otimes \mathbf{H}_t, \tag{4.13}$$

where $\mathbf{H}_{\xi} \in \mathbb{R}^{n_{\xi} \times m_{\xi}}$ is:

$$H_{\xi,ij} = h_{\xi,j}(\boldsymbol{\xi}_i),$$

$\mathbf{H}_s \in \mathbb{R}^{n_s \times m_s}$ is:

$$H_{s,ij} = h_{s,j}(\mathbf{x}_{s,i}),$$

and $\mathbf{H}_t \in \mathbb{R}^{n_t \times m_t}$ is:

$$H_{t,ij} = h_{t,j}(t_i).$$

Efficient predictions and inference Given a set of hyper-parameters $\boldsymbol{\theta}$, all the statistics that are required to make predictions or evaluate the posterior of $p(\boldsymbol{\theta})$ can be calculated efficiently by exploiting the properties of the Kronecker product. Its most important property is that various factorizations (e.g. Cholesky or QR) of a matrix formed by Kronecker products is given by the Kronecker products of the factorizations of the individual matrices [90]. Furthermore, matrix-vector multiplications as well as solving linear systems when the matrices forming the Kronecker product are triangular can be carried out without additional memory. Therefore, working consistently with the Cholesky decomposition of the covariance matrices, leads to very efficient computations. All the linear algebra details pertaining to efficient computations with Kronecker products are documented in C.1 and C.2.

The posterior of the hyper-parameters (see Equation 4.9) can now be sampled efficiently via Gibb's sampling [13], as described in Algorithm 4. Each one of the steps can be carried out using MCMC [65, 47]. The prior distributions as well as the proposal distributions for $\boldsymbol{\theta}_\xi, \boldsymbol{\theta}_s$ and $\boldsymbol{\theta}_t$ are given in the next paragraph.

Choice of the covariance function The separable model described in this section requires the specification of three covariance functions $c_\xi(\cdot,\cdot;\boldsymbol{\theta}_\xi), c_s(\cdot,\cdot;\boldsymbol{\theta}_s)$ and $c_t(\cdot,\cdot;\boldsymbol{\theta}_t)$. In this work, we choose to work with:

$$
\begin{aligned}
c_\xi(\boldsymbol{\xi}_{n_1},\boldsymbol{\xi}_{n_2};\boldsymbol{\theta}_\xi) &:= \exp\left\{-\frac{1}{2}\sum_{k=1}^{k_\xi}\left(\frac{(\xi_{n_1,k}-\xi_{n_2,k})^2}{r_{\xi,k}^2}\right)\right\} + g_\xi\delta_{n_1n_2},\\
c_s(\mathbf{x}_{s,n_1},\mathbf{x}_{s,n_2};\boldsymbol{\theta}_s) &:= \exp\left\{-\frac{1}{2}\sum_{k=1}^{k_s}\left(\frac{(x_{s,n_1,k}-x_{s,n_2,k})^2}{r_{s,k}^2}\right)\right\} + g_s\delta_{n_1n_2},\\
c_t(t_{n_1},t_{n_2};\boldsymbol{\theta}_t) &:= \exp\left\{-\frac{1}{2}\left(\frac{(t_{n_1}-t_{n_2})^2}{r_t^2}\right)\right\} + g_t\delta_{n_1n_2},
\end{aligned}
$$

Algorithm 4: *Sampling the posterior distribution.*

Require: Observed data $\mathbf{X}$ and $\mathbf{Y}$ and initial $\boldsymbol{\theta} = (\boldsymbol{\theta}_\xi, \boldsymbol{\theta}_s, \boldsymbol{\theta}_t)$.

Ensure: Repeated application ensures that $\boldsymbol{\theta} = (\boldsymbol{\theta}_\xi, \boldsymbol{\theta}_s, \boldsymbol{\theta}_t)$ is a sample from Equation 4.9.

Sample:

$$\boldsymbol{\theta}_\xi \sim p(\boldsymbol{\theta}_\xi|\mathbf{Y}, \boldsymbol{\theta}_s, \boldsymbol{\theta}_t) \propto \pi(\boldsymbol{\theta}_\xi)|\mathbf{A}_\xi|^{-\frac{qn}{2n_\xi}}|\mathbf{H}_\xi^T\mathbf{A}_\xi^{-1}\mathbf{H}_\xi|^{-\frac{qm}{2m_\xi}}|\hat{\boldsymbol{\Sigma}}|^{-\frac{n-m}{2}}.$$

Sample:

$$\boldsymbol{\theta}_s \sim p(\boldsymbol{\theta}_s|\mathbf{Y}, \boldsymbol{\theta}_\xi, \boldsymbol{\theta}_t) \propto \pi(\boldsymbol{\theta}_s)|\mathbf{A}_s|^{-\frac{qn}{2n_s}}|\mathbf{H}_s^T\mathbf{A}_s^{-1}\mathbf{H}_s|^{-\frac{qm}{2m_s}}|\hat{\boldsymbol{\Sigma}}|^{-\frac{n-m}{2}}.$$

Sample:

$$\boldsymbol{\theta}_t \sim p(\boldsymbol{\theta}_t|\mathbf{Y}, \boldsymbol{\theta}_\xi, \boldsymbol{\theta}_s) \propto \pi(\boldsymbol{\theta}_t)|\mathbf{A}_t|^{-\frac{qn}{2n_t}}|\mathbf{H}_t^T\mathbf{A}_t^{-1}\mathbf{H}_t|^{-\frac{qm}{2m_t}}|\hat{\boldsymbol{\Sigma}}|^{-\frac{n-m}{2}}.$$

with the hyper-parameters completely specified by:

$$\boldsymbol{\theta}_\xi = (\mathbf{r}_\xi, g_\xi), \boldsymbol{\theta}_s = (\mathbf{r}_s, g_s) \text{ and } \boldsymbol{\theta}_t = (r_t, g_t).$$

The core part of the covariance functions is based on the Square Exponential (SE) kernels with the $\mathbf{r}_\alpha, \alpha = \boldsymbol{\xi}, s, t$ hyper-parameters being interpreted as the length scale of each input dimension. The $g_\alpha, \alpha = \boldsymbol{\xi}, s, t$ are termed "nuggets". The main purpose of the nuggets is to ensure the well-conditioning of the covariance matrices involved in the calculations and they are expected to be typically small (of the order of 10^{-2}). By looking at the full covariance function of the separable model and ignoring second-order products of the nuggets, we can see that the $g_\xi + g_s + g_t$ can be interpreted as the measurement noise. Apart from improving the stability of the computations, one can argue that the presence of the nugget can also lead to better predictive accuracy [40].

The priors of the hyper-parameters $\mathbf{r}_{\xi}, g_{\xi}, \mathbf{r}_s, g_s, r_t$ and g_t, should be chosen to represent any prior knowledge about the computer code that might be available. In order to ensure positive support, we make the common choice $\alpha = \boldsymbol{\xi}, s$ and t:

$$\begin{aligned} \pi(\mathbf{r}_{\alpha,k}|\gamma_\alpha) &= \mathcal{E}(\mathbf{r}_{\alpha,k}|\gamma_\alpha), & (4.14) \\ \pi(g_\alpha|\zeta_\alpha) &= \mathcal{E}(g_\alpha|\zeta_\alpha), & (4.15) \end{aligned}$$

where $\mathcal{E}(\cdot|\lambda)$ denotes the probability density of the exponential distribution with parameter $\lambda > 0$.

For the proposal required by the MCMC sampling schemes described in the previous section, we use a log-normal random walk for all hyper-parameters (again because they are all positive). The step size of the random walk is selected so that the observed acceptance ratio of the MCMC is between 30 and 60 per cent.

The particular values of γ_α and ζ_α for $\alpha = \boldsymbol{\xi}, s$ and t are specified in each numerical example.

4.2.3 Application to uncertainty quantification

In uncertainty quantification tasks, one specifies a probability density on the inputs $\boldsymbol{\xi}$'s, $p(\boldsymbol{\xi})$, and tries to quantify the probability measure induced by it on the output field. In this work, we quantify this uncertainty by interrogating the surrogate built using the Gaussian process model introduced in the previous sections (see Equation 4.8). The whole process is complicated by the fact that our model in reality defines a probability measure over the function space of potential surrogates. This probability measure essentially quantifies the lack of information regarding the real response due to the finite number of observations. In a fully Bayesian setting, this probability measure will be reflected as a probability measure on the predicted statistics (e.g. mean, variance, PDFs, etc.). To the best of our knowledge, such ideas were introduced for the first time in the statistics literature [71] but were largely ignored by the UQ community. Inspired by the above mentioned work, we will describe in this section how our model can be used to essentially sample the posterior distribution of the induced statistics. The procedure is conceptually simple and described in Algorithm 5. The key component of this algorithm is the ability to sample a response surface based on Equation 4.8 that can be described analytically via a kernel representation. This is achieved through the generalization of the techniques discussed in [71]. The final component of the algorithm has to do with the evaluation of the statistics of interest induced by this response surface. We will show that our model allows for semi-analytic calculation of all statistics up to second-order. Higher-order statistics, or full probability densities have to be obtained using Monte Carlo techniques on the sampled surrogate surface.

Algorithm 5: *Sampling the posterior of the statistics. By repeatedly calling this algorithm, error bars for the desired statistics may be obtained.*

Require: Observed data $\mathbf{X}$ and $\mathbf{Y}$ and $\boldsymbol{\theta}_0$ sample from Equation 4.9.

Ensure: $\mathcal{S}$ is a sample from the statistic of interest.

Sample a new $\boldsymbol{\theta}_1$ following the Gibb's procedure given in Section 4.2.2.

Sample a response surface using Algorithm 6.

Interrogate the obtained response surface (analytically or via MC) to obtain $\mathcal{S}$.

Sampling a response surface In order to obtain an analytical representation of the response surface, reference [71] suggests selecting a space filling design of the input variables, using Equation 4.8 to sample the outputs and then augmenting the original data set with the new observations to derive an updated Equation 4.8 with reduced variance. The mean of the updated posterior predictive distribution is an analytic function that can be thought of (if its variance is sufficiently small) as a sample from the predictive probability measure. Several problems arise if one follows this approach. To start with, one does not know a priori how many design points are required in order to reduce the predictive variance to a pre-specified tolerance. Furthermore, design points must be well placed and far away from the initial observations in order to avoid numerical instabilities. Finally - and this is particular to our model - including design points in all sets of different inputs ($\boldsymbol{\xi}$, $\mathbf{x}_s$ and t) breaks down the Kronecker product representation of the covariance and design matrices which, in turn, leads to a tremendous computational burden. In order to avoid the latter of these conundrums, we choose to work with the same spatial and time points as the ones included in the original data (namely $\mathbf{X}_s$ and $\mathbf{X}_t$). This approximation, will ignore only a - hopefully - small part of the epistemic uncertainty due to the finite number of observations. That is, we only choose design points in $\mathcal{X}_{\xi}$. The former two problems are addressed by employing a sequential strategy in which the $\boldsymbol{\xi}$'s are selected one by one by maximizing the predictive variance until a specific tolerance is achieved. This approach is guaranteed to produce a space filling design that is well separated from the original observations. In addition, the only covariance matrix that needs to be updated is the one pertaining to $\boldsymbol{\xi}$. In C.3, we describe how the Cholesky decomposition of the covariance matrix as well as solutions to the relevant linear systems can be updated in quadratic time when new design points are added.

Consider $\boldsymbol{\theta}$ fixed and let $\mathbf{X}_{\xi,d} \in \mathbb{R}^{(n_\xi+d)\times k_\xi}$ and $\mathbf{Y}_d \in \mathbb{R}^{((n_\xi+d)n_s n_t)\times q}$ denote the set of $\boldsymbol{\xi}$'s and the corresponding outputs when d design points have been observed. That is

$$\mathbf{X}_{\xi,d} = \begin{pmatrix} \mathbf{X}_\xi \\ \boldsymbol{\xi}_{n_\xi+1} \\ \vdots \\ \boldsymbol{\xi}_{n_\xi+d} \end{pmatrix}.$$

For $d = 0$, we obtain the observed data:

$$\mathbf{X}_{\xi,0} = \mathbf{X}_\xi \text{ and } \mathbf{Y}_0 = \mathbf{Y}.$$

Define $\hat{\mathbf{B}}_d \in \mathbb{R}^{m\times q}$, $\mathbf{H}_{\xi,d} \in \mathbb{R}^{(n_\xi+d)\times m_\xi}$, $\mathbf{A}_{\xi,d} \in \mathbb{R}^{(n_\xi+d)\times(n_\xi+d)}$ to be the weight, design and covariance matrices pertaining to $\boldsymbol{\xi}$, respectively, when $\mathbf{X}_{\xi,d}$ and $\mathbf{Y}_d$ have been observed. In order to avoid cluttering the final formulas, let us also define:

$$\mathbf{a}_{\xi,d}(\boldsymbol{\xi}) = \begin{pmatrix} \mathbf{a}_\xi \\ c_\xi(\boldsymbol{\xi}_{n_\xi+1}, \boldsymbol{\xi}; \boldsymbol{\theta}) \\ \vdots \\ c_\xi(\boldsymbol{\xi}_{n_\xi+d}, \boldsymbol{\xi}; \boldsymbol{\theta}) \end{pmatrix} \in \mathbb{R}^{n_\xi+d},$$

$$\mathbf{A}_d = \mathbf{A}_{\xi,d} \otimes \mathbf{A}_s \otimes \mathbf{A}_t,$$

and

$$\mathbf{H}_d = \mathbf{H}_{\xi,d} \otimes \mathbf{H}_s \otimes \mathbf{H}_t.$$

Now, let $\boldsymbol{\xi} \in \mathcal{X}_\xi$ and $\mathbf{Z}(\boldsymbol{\xi}) \in \mathbb{R}^{(n_s n_t)\times q}$ be the output at $\boldsymbol{\xi}$ and all spatial and time points in $\mathbf{X}_s$ and $\mathbf{X}_t$. By using Bayes theorem and Equation 4.8, we can show that:

$$\mathbf{Z}(\boldsymbol{\xi}) | \mathbf{Y}_d, \boldsymbol{\theta} \sim \mathcal{T}_{(n_s n_t)\times q}\left(\mathbf{M}_d(\boldsymbol{\xi}), \mathbf{C}_d(\boldsymbol{\xi}), \hat{\boldsymbol{\Sigma}}; n_d - m\right), \tag{4.16}$$

where $n_d = (n_\xi + d)n_s n_t$ and the mean is given by:

$$\begin{aligned}\mathbf{M}_d(\boldsymbol{\xi}) &= \left(\mathbf{h}_\xi^T(\boldsymbol{\xi}) \otimes \mathbf{H}_s \otimes \mathbf{H}_t\right)\hat{\mathbf{B}}_d \\ &\quad + \left(\mathbf{a}_{\xi,d}(\boldsymbol{\xi})^T \otimes \mathbf{A}_s \otimes \mathbf{A}_t\right)\mathbf{A}_d^{-1}\left(\mathbf{Y}_d - \mathbf{H}_d\hat{\mathbf{B}}_d\right),\end{aligned} \tag{4.17}$$

and the covariance matrix by:

$$\begin{aligned}\mathbf{C}_d(\boldsymbol{\xi}) &= c_\xi(\boldsymbol{\xi}, \boldsymbol{\xi}; \boldsymbol{\theta})\left(\mathbf{A}_s \otimes \mathbf{A}_t\right) \\ &\quad - \left(\mathbf{a}_{\xi,d}(\boldsymbol{\xi}) \otimes \mathbf{A}_s \otimes \mathbf{A}_t\right)^T \mathbf{A}_d^{-1}\left(\mathbf{a}_{\xi,d}(\boldsymbol{\xi}) \otimes \mathbf{A}_s \otimes \mathbf{A}_t\right) \\ &\quad \left(\left(\mathbf{h}_\xi(\boldsymbol{\xi}) \otimes \mathbf{H}_s \otimes \mathbf{H}_t\right) - \mathbf{H}_d^T\mathbf{A}_d^{-1}\right)^T\left(\mathbf{H}_d^T\mathbf{A}_d^{-1}\mathbf{H}_d\right)^{-1} \cdot \\ &\quad \left(\left(\mathbf{h}_\xi(\boldsymbol{\xi}) \otimes \mathbf{H}_s \otimes \mathbf{H}_t\right) - \mathbf{H}_d^T\mathbf{A}_d^{-1}\right).\end{aligned} \tag{4.18}$$

In order to sample Equation 4.16, we need to compute the Cholesky decomposition of $\mathbf{C}_d(\boldsymbol{\xi})$. This is not trivial, since $\mathbf{C}_d(\boldsymbol{\xi})$ does not have a particular structure. In the numerical examples - and in particular for the porous flow problem considered in Section 3.3.3- this matrix turned out to be extremely ill-conditioned. Even though, theoretically $\mathbf{C}_d(\boldsymbol{\xi})$ is guaranteed to be symmetric positive definite, numerically it must be treated as positive semi-definite. For this reason, one has to use a low-rank approximation of $\mathbf{C}_d(\boldsymbol{\xi})$ using the pivoted Cholesky factorization [45]. This can be carried out using the LAPACK routine dpstrf. The tolerance we used for this approximation was 10^{-3} for all numerical examples we considered. We found no observable difference between samples obtained with the normal Cholesky factorization and this approach. Finally, let us mention that a scalar quantity that is associated directly with the uncertainty pertaining $\boldsymbol{\xi}$ is given by:

$$\sigma_d^2(\boldsymbol{\xi}) = \frac{\mathrm{tr}[\mathbf{C}_d(\boldsymbol{\xi})]\,\mathrm{tr}[\hat{\boldsymbol{\Sigma}}]}{n_s n_t q} p(\boldsymbol{\xi}). \tag{4.19}$$

This is the sum of the variances of all outputs at all different spatial and time points weighted by the input probability distribution $p(\boldsymbol{\xi})$. The idea is to sequentially augment the data set by including the design points from a dense subset $\mathcal{X}_{\xi}^{*}$ of $\mathcal{X}_{\xi}$ that maximize Equation 4.19 until a pre-specified tolerance is achieved. At that point, the joint predictive mean given by Equation 4.17 may be used as an analytic sample surrogate surface. In general, we would like to evaluate the response surface on a denser spatial design $\mathbf{X}_s^* \in \mathbb{R}^{n_s^* \times k_s}$ and/or more time steps $\mathbf{X}_t^* \in \mathbb{R}^{n_t^*}$). The joint predictive mean at those points is given by:

$$\begin{aligned}\mathbf{M}_d^*(\boldsymbol{\xi}) \quad = \quad & \left(\mathbf{h}_{\xi}(\boldsymbol{\xi})^T \otimes \mathbf{H}_s^* \otimes \mathbf{H}_t^*\right) \hat{\mathbf{B}}_d \\ & + \left(\mathbf{a}_{\xi,d}(\boldsymbol{\xi})^T \otimes \mathbf{A}_s^{*,T} \otimes \mathbf{A}_t^{*,T}\right) \mathbf{A}_d^{-1} \left(\mathbf{Y}_d - \mathbf{H}_d \hat{\mathbf{B}}_d\right),\end{aligned} \tag{4.20}$$

where $\mathbf{H}_s^* \in \mathbb{R}^{n_s^* \times m_s}, \mathbf{H}_t^* \in \mathbb{R}^{n_t^* \times m_t}$ are the design matrices that pertain to the test spatial and time points, respectively, while $\mathbf{A}_s^* \in \mathbb{R}^{n_s \times n_s^*}, \mathbf{A}_t^* \in \mathbb{R}^{n_t \times n_t^*}$ are the corresponding cross covariance matrices. We identify Equation 4.20 as a sample response surface from the function space of possible surrogates. The complete algorithmic details are given in Algorithm 6.

Analytic first-order and second-order statistics In applications, we are usually interested in first and second-order statistics. We can obtain a sample of the mean response by integrating out $\boldsymbol{\xi}$ from Equation 4.20:

$$\mathbf{M}_d^* := \int \mathbf{M}_d^*(\boldsymbol{\xi}) p(\boldsymbol{\xi}) d\boldsymbol{\xi}. \tag{4.22}$$

It can be easily shown that:

$$\begin{aligned}\mathbf{M}_d^* \quad = \quad & \left(\boldsymbol{\epsilon}_{\mathbf{h}}^T \otimes \mathbf{H}_s^* \otimes \mathbf{H}_t^*\right) \hat{\mathbf{B}}_d \\ & + \left(\boldsymbol{\epsilon}_{\mathbf{a},d}^T \otimes \mathbf{A}_s^{*,T} \otimes \mathbf{A}_t^{*,T}\right) \mathbf{A}_d^{-1} \left(\mathbf{Y}_d - \mathbf{H}_d \hat{\mathbf{B}}_d\right),\end{aligned} \tag{4.23}$$

where

$$\boldsymbol{\epsilon}_{\mathbf{h}} = \int \mathbf{h}_{\xi}(\boldsymbol{\xi}) p(\boldsymbol{\xi}) d\boldsymbol{\xi} \;\text{ and }\; \boldsymbol{\epsilon}_{\mathbf{a},d} = \int \mathbf{a}_{\xi,d}(\boldsymbol{\xi}) p(\boldsymbol{\xi}) d\boldsymbol{\xi}.$$

Now, let $i, j \in \{1, \dots, q\}$ be two arbitrary outputs. The covariance matrix between all possible spatial and time test points is defined by:

$$\mathbf{C}_d^{ij,*} := \int \left(\mathbf{M}_{d,i}^*(\boldsymbol{\xi}) - \mathbf{M}_{d,i}^*\right)\left(\mathbf{M}_{d,j}^*(\boldsymbol{\xi}) - \mathbf{M}_{d,j}^*\right)^T p(\boldsymbol{\xi}) d\boldsymbol{\xi}, \tag{4.24}$$

Algorithm 6: *Sample a response surface.*

Require: Observed data $\mathbf{X}$ and $\mathbf{Y}$, $\boldsymbol{\theta}$ sampled from Equation 4.8, a dense set of design points $\mathcal{X}_{\xi}^{*} = \{\boldsymbol{\xi}_1^*, \dots, \boldsymbol{\xi}_{n_\xi^*}^*\}$, the desired final tolerance $\delta > 0$ and dense spatial and time designs $\mathbf{X}_s^* \in \mathbb{R}^{n_s^* \times k_s}$ and $\mathbf{X}_t^* \in \mathbb{R}^{n_t^*}$ on which we wish to make predictions.

Ensure: After $d \geq 1$ steps, the uncertainty of Equation 4.16 as captured by Equation 4.19 is less than δ and $\mathbf{M}_d^*(\boldsymbol{\xi})$ given by Equation 4.20 can be used as an analytic representation of the sampled response surface.

Initialize $d \leftarrow 0$.

repeat

Find the next design point:

$$\boldsymbol{\xi}_{n_\xi+d+1} = \arg\max_{\boldsymbol{\xi} \in \mathcal{X}_\xi^*} \sigma_d^2(\boldsymbol{\xi}). \tag{4.21}$$

Sample $\mathbf{Z}(\boldsymbol{\xi}_{n_\xi+d+1})$ from Equation 4.16.

Augment the set of observations with the pair $\left(\boldsymbol{\xi}_{n_\xi+d+1}, \mathbf{Z}(\boldsymbol{\xi}_{n_\xi+d+1})\right)$.

$d \leftarrow d + 1$.

until $\sigma_d^2(\boldsymbol{\xi}_{n_\xi+d}) < \delta$.

where the subscripts i and j select columns of the associated matrices. This matrix, contains all second-order statistics of the surrogate. For example, the variance of each output $i = 1, \dots, q$ is on all spatial and time locations $\mathbf{X}_s^*$ and $\mathbf{X}_t^*$, respectively is given by:

$$\mathbf{V}_d^{i,*} := \operatorname{diag} \mathbf{C}_d^{ii,*}. \tag{4.25}$$

It can be shown using tensorial notation, that $\mathbf{C}_d^{ij,*}$ may be evaluated by:

$$
\begin{aligned}
\mathbf{C}_d^{ij,*} \quad = \quad & \left(\mathbf{H}_s^* \otimes \mathbf{H}_t^*\right) \hat{\mathbf{B}}_d^i \boldsymbol{\nu}_{\mathbf{hh}} \left[\left(\mathbf{H}_s^* \otimes \mathbf{H}_t^*\right) \hat{\mathbf{B}}_d^j\right]^T \\
& + \left(\mathbf{A}_s^{*,T} \otimes \mathbf{A}_t^{*,T}\right) \tilde{\mathbf{Y}}_d^i \boldsymbol{\nu}_{\mathbf{aa},d} \left[\left(\mathbf{A}_s^{*,T} \otimes \mathbf{A}_t^{*,T}\right) \tilde{\mathbf{Y}}_d^j\right]^T \\
& + \left(\mathbf{H}_s^* \otimes \mathbf{H}_t^*\right) \hat{\mathbf{B}}_d^i \boldsymbol{\nu}_{\mathbf{ha},d} \left[\left(\mathbf{A}_s^{*,T} \otimes \mathbf{A}_t^{*,T}\right) \tilde{\mathbf{Y}}_d^j\right]^T \\
& + \left(\mathbf{A}_s^{*,T} \otimes \mathbf{A}_t^{*,T}\right) \tilde{\mathbf{Y}}_d^i \boldsymbol{\nu}_{\mathbf{ah},d} \left[\left(\mathbf{H}_s^* \otimes \mathbf{H}_t^*\right) \hat{\mathbf{B}}_d^j\right]^T,
\end{aligned}
\tag{4.26}
$$

where $\hat{\mathbf{B}}_d^i \in \mathbb{R}^{(m_s m_t)\times m_\xi}$ is such that $\mathrm{vec}(\hat{\mathbf{B}}_d^i) = \hat{\mathbf{B}}_{d,i}$ (i.e. the i-th column of $\mathbf{B}_d$), $\tilde{\mathbf{Y}}_d \in \mathbb{R}^{n\times q}$ is defined by:

$$
\tilde{\mathbf{Y}}_d = \mathbf{A}_d^{-1}\left(\mathbf{Y}_d - \mathbf{H}\hat{\mathbf{B}}_d\right),
$$

$\tilde{\mathbf{Y}}_d^i \in \mathbb{R}^{(n_s n_t)\times n_\xi}$ is such that $\mathrm{vec}(\tilde{\mathbf{Y}}_d^i) = \tilde{\mathbf{Y}}_{d,i}$, $\boldsymbol{\nu}_{\mathbf{hh}} \in \mathbb{R}^{m_\xi \times m_\xi}$ is given by:

$$
\boldsymbol{\nu}_{\mathbf{hh}} = \int \left(\mathbf{h}_\xi(\boldsymbol{\xi}) - \boldsymbol{\epsilon}_{\mathbf{h}}\right)\left(\mathbf{h}_\xi(\boldsymbol{\xi}) - \boldsymbol{\epsilon}_{\mathbf{h}}\right)^T p(\boldsymbol{\xi}) d\boldsymbol{\xi}, \tag{4.27}
$$

$\boldsymbol{\nu}_{\mathbf{aa},d} \in \mathbb{R}^{n_\xi \times n_\xi}$ by:

$$
\boldsymbol{\nu}_{\mathbf{aa},d} = \int \left(\mathbf{a}_{\xi,d}(\boldsymbol{\xi}) - \boldsymbol{\epsilon}_{d,\mathbf{a}}\right)\left(\mathbf{a}_{\xi,d}(\boldsymbol{\xi}) - \boldsymbol{\epsilon}_{\mathbf{a},d}\right)^T p(\boldsymbol{\xi}) d\boldsymbol{\xi}, \tag{4.28}
$$

$\boldsymbol{\nu}_{\mathbf{ha},d} \in \mathbb{R}^{m_\xi \times n_\xi}$ by:

$$
\boldsymbol{\nu}_{\mathbf{ha},d} = \int \left(\mathbf{h}_\xi(\boldsymbol{\xi}) - \boldsymbol{\epsilon}_{\mathbf{h}}\right)\left(\mathbf{a}_{\xi,d}(\boldsymbol{\xi}) - \boldsymbol{\epsilon}_{\mathbf{a},d}\right)^T p(\boldsymbol{\xi}) d\boldsymbol{\xi}, \tag{4.29}
$$

and $\boldsymbol{\nu}_{\mathbf{ah},d} = \boldsymbol{\nu}_{\mathbf{ha},d}^T$.

4.3 Numerical Examples

4.3.1 Krainchnan-Orszag three-mode problem

Consider the system of ordinary differential equations [91]:

$$\begin{aligned}\frac{dy_1}{dt} &= y_1 y_3,\\ \frac{dy_2}{dt} &= -y_2 y_3,\\ \frac{dy_3}{dt} &= -y_1^2 + y_2^2,\end{aligned}$$

subject to random initial conditions at $t = 0$: The stochastic initial conditions are defined by:

$$y_1(0) = 1,\ y_2(0) = 0.1\xi_1,\ y_3(0) = \xi_2,$$

where

$$\xi_i \sim \mathcal{U}([-1, 1]),\ i = 1, 2.$$

This dynamical system is particularly interesting because the response has a discontinuity at $\xi_1 = 0$. The deterministic solver we use is a 4-th order Runge-Kutta method as implemented in GNU Scientific Library [30].

As is apparent, the input variables $\boldsymbol{\xi}$ represent the initial conditions. We will consider the case of two input dimensions, i.e. $k_\xi = 2$. The output consist of three distinct variables ($q = 3$) that are functions of time ($k_s = 0$). For convenience, we choose to work with a constant prior mean by selecting:

$$\mathbf{h}_\xi(\boldsymbol{\xi}) = 1 \text{ and } \mathbf{h}_t(t) = 1.$$

That is, $m_\xi = 1, m_t = 1$. We fix n_ξ and gather the input data $\mathbf{X}_\xi \in \mathbb{R}^{n_\xi \times k_\xi}$ from a Latin hyper-cube design [**?**]. We solve the system for the time interval $[0, 10]$

and record the response at 20 equidistant time steps, i.e. $\mathbf{X}_t \in \mathbb{R}^{n_t}$ with $n_t = 20$. Both $\mathbf{X}_{\xi}$ and $\mathbf{X}_t$ are scaled in $[0, 1]$. The priors are specified by selecting:

$$\gamma_\alpha = 1/0.05 \text{ and } \zeta_\alpha = 10^6, \text{ for } \alpha = \boldsymbol{\xi}, t.$$

This means, that we a priori assume that the mean for all length scales is 0.05 and the mean of all the nuggets is 10^{-6}. We train our model for $n_{\xi} = 70, 100$ and 150 observations by sampling the posterior of $\boldsymbol{\theta} = (\mathbf{r}_{\xi}, g_{\xi}, r_t, g_t)$ given in Equation 4.9 following the Gibbs-MCMC procedure described in Algorithm 4. To initialize the Markov chain, we sample the prior (Eqs. 4.14 and 4.15) of the hyper-parameters 100 times and set $\boldsymbol{\theta}_0$ equal to the sample with maximum posterior probability defined by Equation 4.9. The proposals are selected to be log-normal random walks and the step size (the same for all types of inputs) is set to 0.01. The chain is well mixed after about 500 iterations of the Gibb's scheme.

After the Markov chain has been sufficiently mixed, we are ready to start making predictions. Predictions are made at 50 equidistant time steps in $[0, 10]$, i.e. $\mathbf{X}_t^* \in \mathbb{R}^{n_t^*}$ with $n_t^* = 50$. Then, we draw 100 samples from the posterior distribution of the statistics of interest as described in Algorithm 5 with tolerance $\delta = 10^{-2}$. We plot the mean of the statistics as well as 95% error bars (2 times the standard deviation of the statistic). To calculate the mean of a sampled response surface, we use Equation 4.23 while for the variance we use the diagonal of $\mathbf{C}_d^{ii,*}, i = 1, \dots, q$ (Equation 4.26). One or two dimensional probability densities for each sampled response surface are evaluated by the following MC procedure: (1) We draw 10,000 samples from $p(\boldsymbol{\xi})$; (2) We evaluate the sampled response (Equation 4.20) at each one of these $\boldsymbol{\xi}$'s; (3) We use a one- or two-dimensional kernel density estimator [10] to approximate the desired PDF. The predicted means of all the statistics are practically identical to the ones obtained via Monte Carlo estimate (not shown in the figures, see [**?**]). The first row of

Figure 4.1 shows the time evolution of the mean of $y_1(t)$ and $y_3(t)$ for $n_\xi = 100$. Notice that the error bars are very tight. The second and third rows of the same figures depict the variance of the same quantities for $n_\xi = 100$ and $n_\xi = 150$, respectively. We can see the width of the error bars decreasing as the number of observations is increased. Figure 4.2 shows the time evolution of the probability density of $y_2(t)$. The four rows correspond to different time instants (specifically $t = 4, 6, 8$ and 10). The columns correspond to $n_\xi = 70, 100$ and 150 counting from the left. Figure 4.3 shows the time evolution of the joint probability density of $y_2(t)$ and $y_3(t)$. The four rows correspond to different time instants (specifically $t = 4, 6, 8$ and 10). The columns correspond to $n_\xi = 70, 100$ and 150 counting from the left.

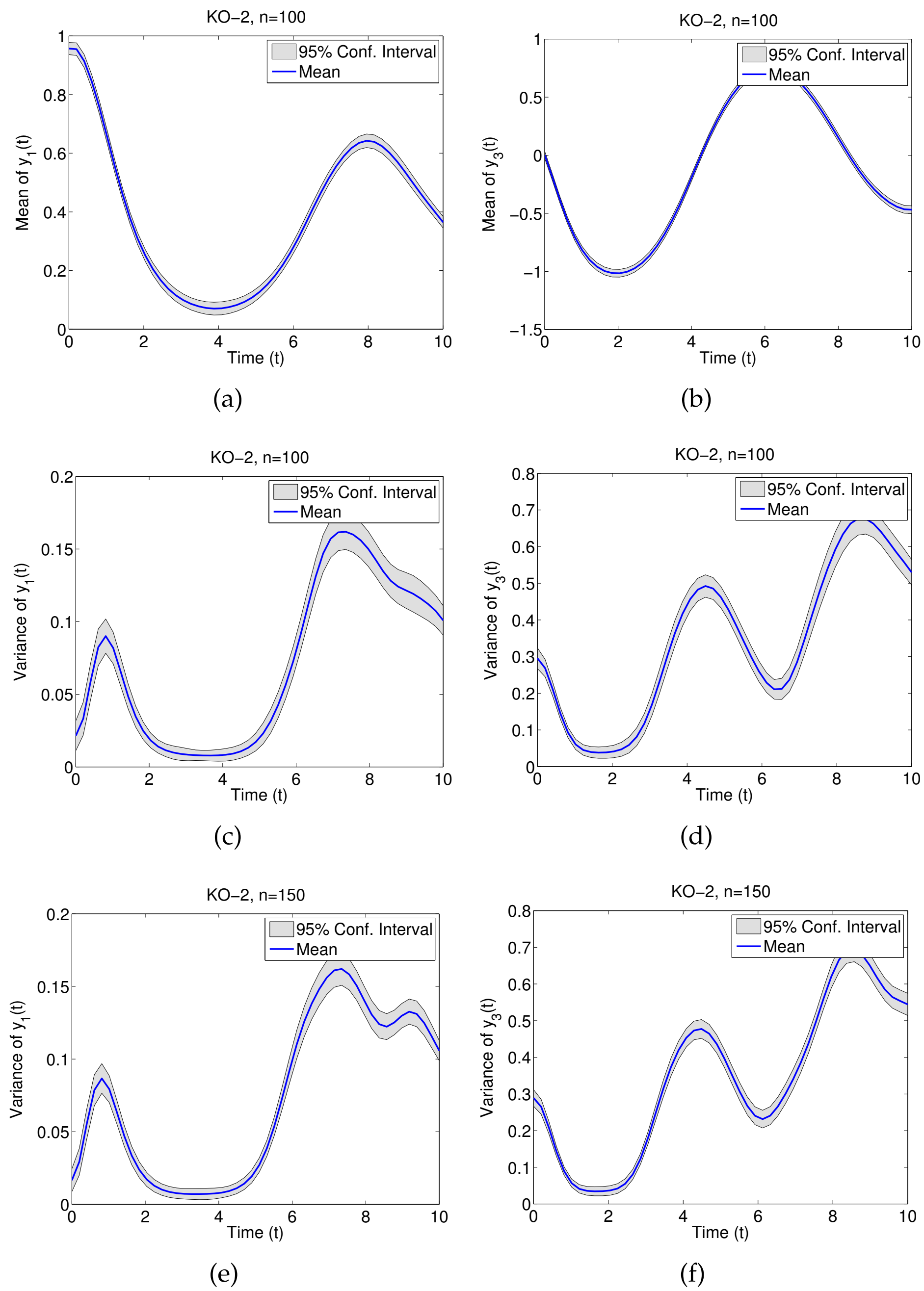


Figure 4.1: KO-2: The thick blue line is the mean of the statistic predicted by our model while the gray area provides 95% confidence intervals. The first row ((a) and (b)) corresponds to the mean of the response as captured with $n_\xi = 100$. The second ((c) and (d)) and the last ((e) and (f)) rows show the variance of the response for $n_\xi = 100$ and $n_\xi = 150$, respectively.

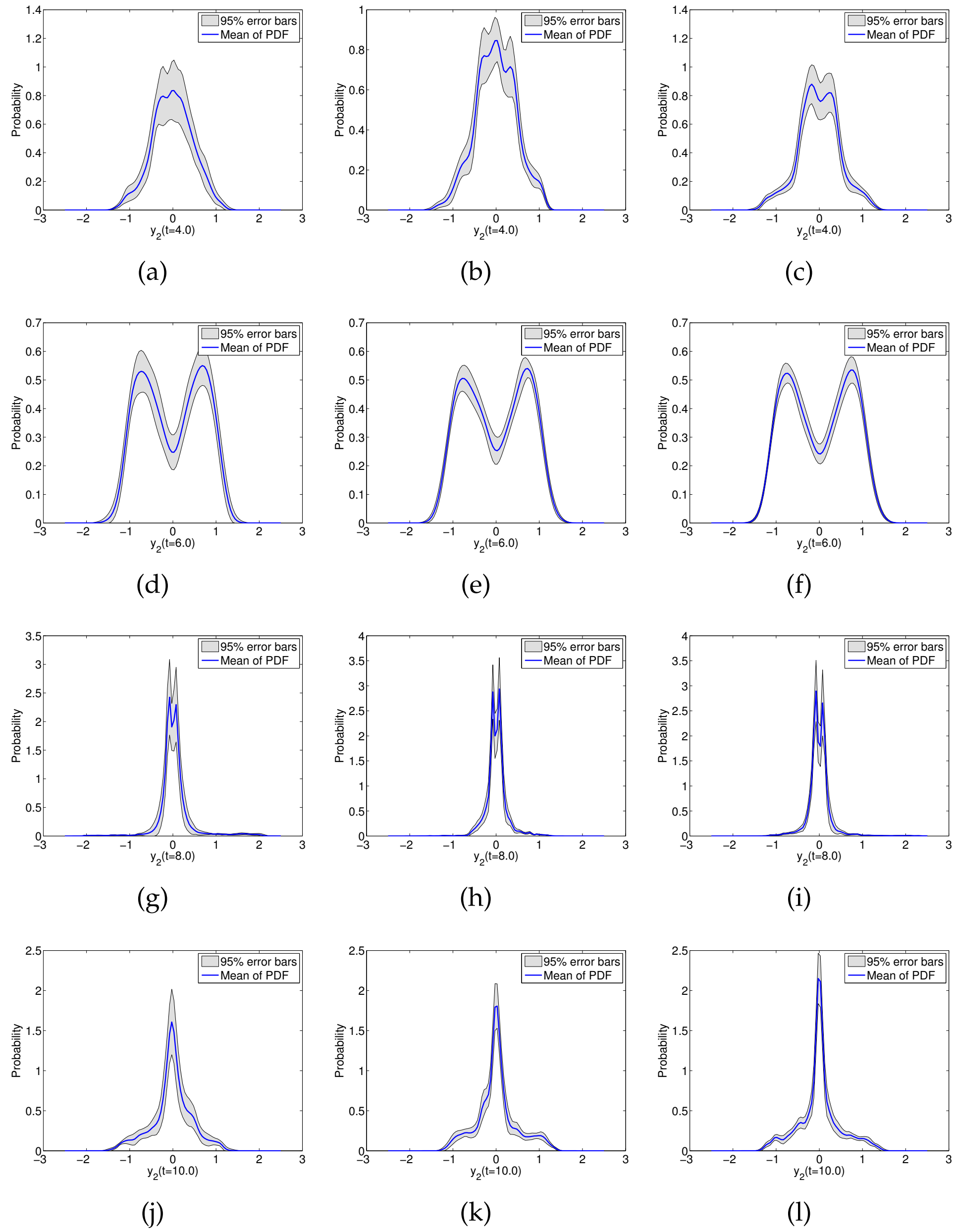


Figure 4.2: KO-2: The first column corresponds to $n_\xi = 70$, the second to $n_\xi = 100$ and the third to $n_\xi = 150$. Each row depicts the PDF of $y_2(t)$ for times $t = 4, 6, 8, 10$. The thick blue line is the mean of the PDF predicted by our model while the gray area provides 95% confidence intervals.

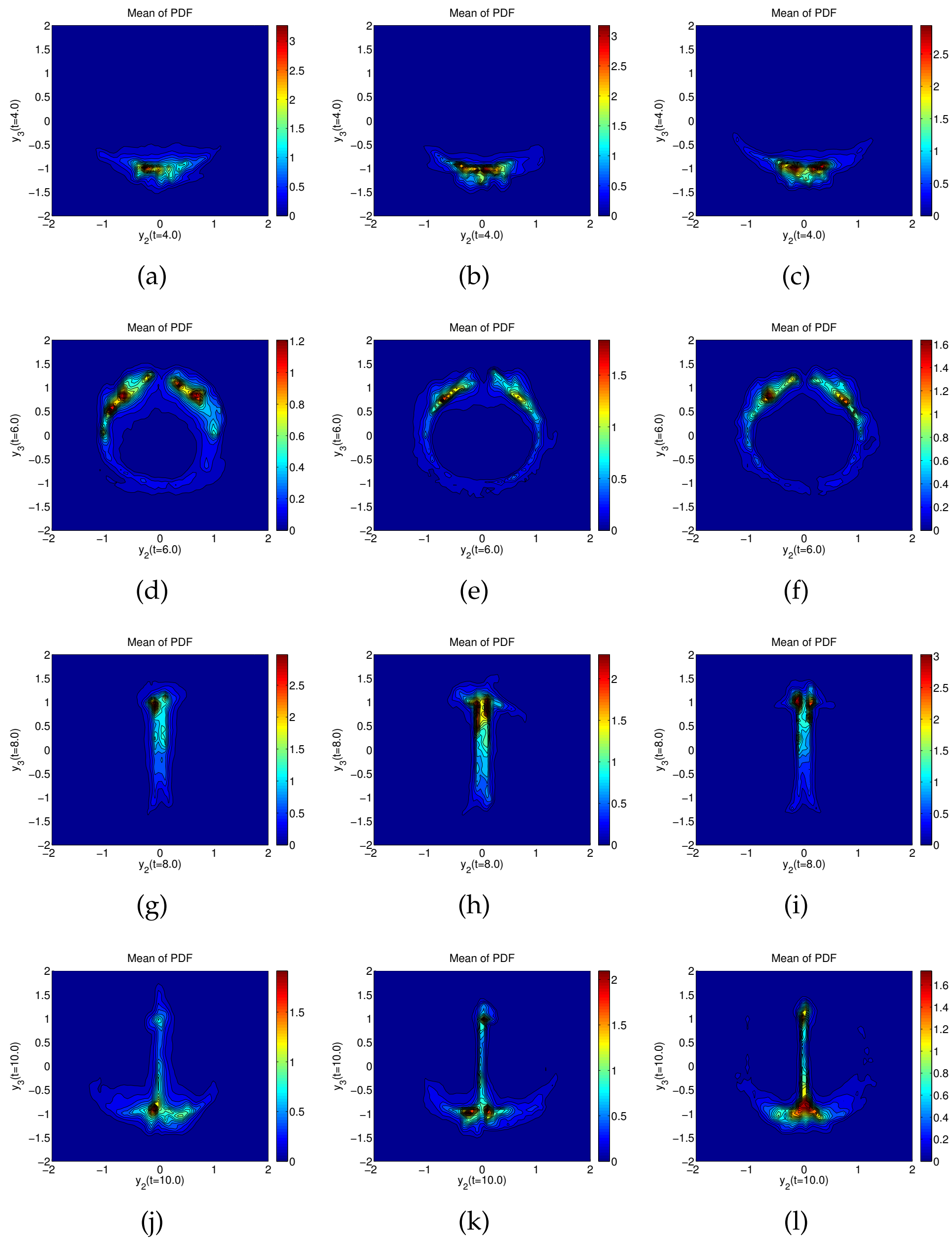


Figure 4.3: KO-2: The first column corresponds to $n_\xi = 70$, the second to $n_\xi = 100$ and the third to $n_\xi = 150$. Each row depicts the joint PDF of $y_2(t)$ and $y_3(t)$ for times $t = 4, 6, 8, 10$.

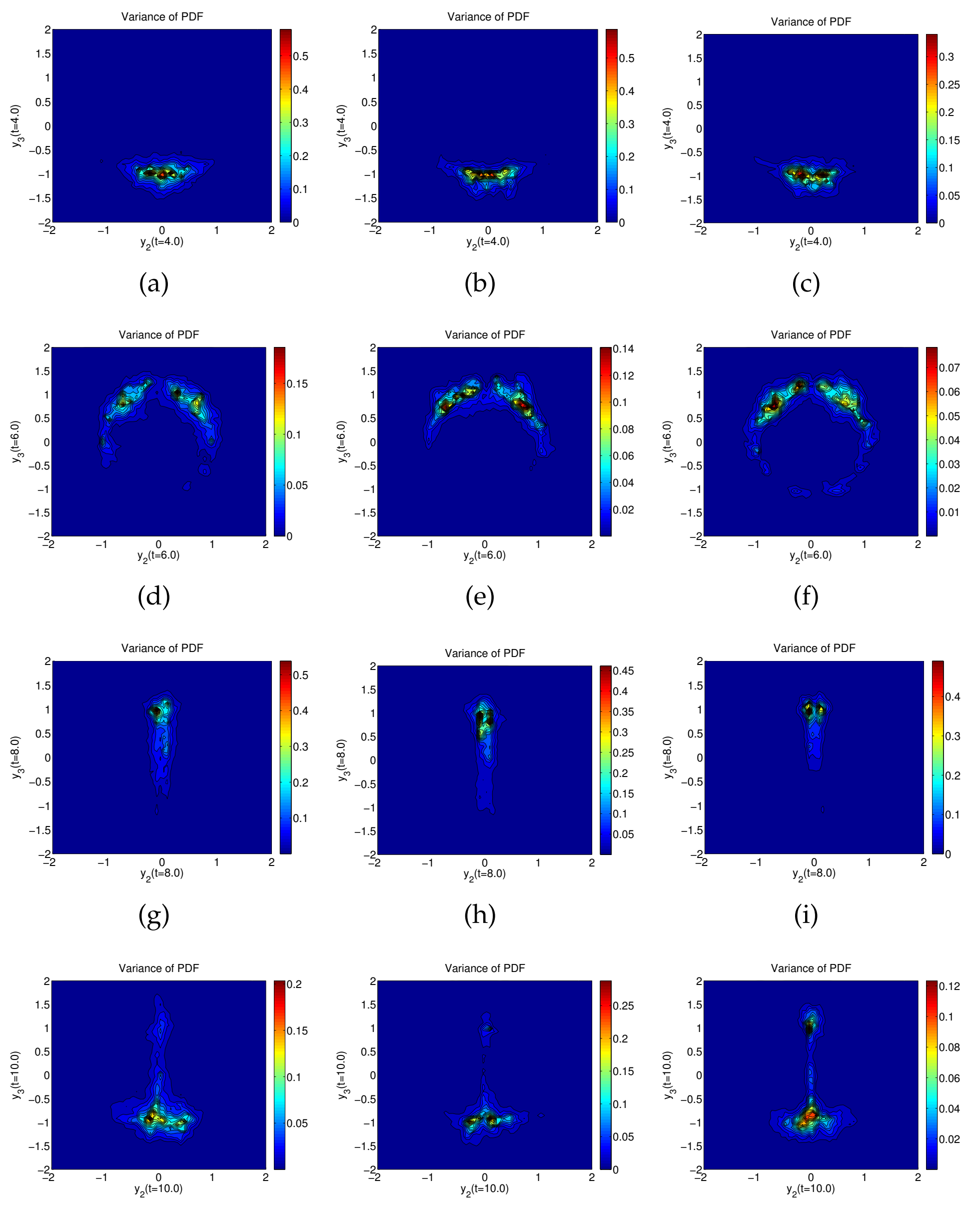


Figure 4.4: KO-2: The first column corresponds to $n_\xi = 70$, the second to $n_\xi = 100$ and the third to $n_\xi = 150$. Each row depicts the predictive variance of the joint PDF of $y_2(t)$ and $y_3(t)$ for times $t = 4, 6, 8, 10$.

4.3.2 Flow through porous media

In this example, we study a two-dimensional, single phase, steady-state flow through a random permeability field. A good review of the mathematical models of flow through porous media can be found in [1]. The spatial domain $\mathcal{X}_s$ is chosen to be the unit square $[0,1]^2$, representing an idealized oil reservoir. Let us denote with p and $\mathbf{u}$ the pressure and the velocity fields of the fluid, respectively. These are connected via the Darcy law:

$$\mathbf{u} = -\mathbf{K}\nabla p, \text{ in } \mathcal{X}_s, \tag{4.30}$$

where $\mathbf{K}$ is the permeability tensor that models the easiness with which the liquid flows through the reservoir. Combining the Darcy law with the continuity equation, it is easy to show that the governing PDE for the pressure is:

$$-\nabla\cdot(\mathbf{K}\nabla p) = f, \text{ in } \mathcal{X}_s, \tag{4.31}$$

where the source term f may be used to model injection/production wells. We use two model square wells: an injection well on the left-bottom corner of $\mathcal{X}_s$ and a production well on the top-right corner. The particular mathematical form is as follows:

$$f(\mathbf{x}_s) = \begin{cases} -r, & \text{if } |x_{si} - \frac{1}{2}w| < \frac{1}{2}w, \text{ for } i = 1,2, \\ r, & \text{if } |x_{si} - 1 + \frac{1}{2}w| < \frac{1}{2}w, \text{ for } i = 1,2, \\ 0, & \text{otherwise}, \end{cases} \tag{4.32}$$

where r specifies the rate of the wells and w their size (chosen to be $r = 10$ and $w = 1/8$). Furthermore, we impose no-flux boundary conditions on the walls of the reservoir:

$$\mathbf{u}\cdot\hat{\mathbf{n}} = 0, \text{ on } \partial\mathcal{X}_s, \tag{4.33}$$

where $\hat{\mathbf{n}}$ is the unit normal vector of the boundary. These boundary conditions specify the pressure p up to an additive constant. To assure uniqueness of the boundary value problem defined by Eqs. (4.30), (4.31) and (4.33), we impose the constraint [8]:

$$\int_{\mathcal{X}_s} p(x)dx = 0.$$

We restrict ourselves to an isotropic permeability tensor:

$$K_{ij} = K\delta_{ij}.$$

K is modeled as

$$K(\mathbf{x}_s) = \exp\left\{G(\mathbf{x}_s)\right\},$$

where G is a Gaussian random field:

$$G(\cdot) \sim \mathcal{N}\left(m, c_G(\cdot,\cdot)\right),$$

with constant mean m and an exponential covariance function given by

$$c_G(\mathbf{x}_{s1}, \mathbf{x}_{s2}) = s_G^2 \exp\left\{-\sum_{k=1}^{k_s} \frac{|x_{s1,k} - x_{s2,k}|}{\ell_k}\right\}. \tag{4.34}$$

The parameters ℓ_k represent the correlation lengths of the field, while $s_G > 0$ is its variance. The values we choose for the parameters are $m = 0$, $\ell_k = 0.1$ and $s_G = 1$. In order to obtain a finite dimensional representation of G, we employ the Karhunen-Loève expansion [34] and truncate it after $k_\xi = 50$ terms:

$$G(\mathbf{w}; \mathbf{x}_s) = m + \sum_{k=1}^{k_\xi} w_k \psi_k(\mathbf{x}_s),$$

where $\mathbf{w} = (w_1, \dots, w_{k_\xi})$ is vector of independent, zero mean and unit variance Gaussian random variables and $\psi_k(\mathbf{x}_s)$ are the eigenfunctions of the exponential covariance given in Equation 4.34 (suitably normalized, of course). In order to guarantee the analytical calculation of statistics of the first-order p and $\mathbf{u}$ of Section 4.2.3, we choose to work with the uniform random variables

$$\xi_k = \Phi(w_k) \sim \mathcal{U}[0,1],\ k = 1,\dots,\mathbf{k}_{\boldsymbol{\xi}},$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution. Putting it all together, the finite-dimensional stochastic representation of the permeability field is:

$$K(\boldsymbol{\xi};\mathbf{x}_s) = \exp\left\{m + \sum_{k=1}^{k_{\xi}} \Phi^{-1}(\xi_k)\psi_k(\mathbf{x}_s)\right\}. \tag{4.35}$$

In order to make the notational connection with the rest of the paper obvious, let us define the response of the physical model as

$$\mathbf{f} : \mathcal{X}_{\xi} \times \mathcal{X}_s \to \mathbb{R}^q,$$

where, of course, $\mathcal{X}_{\xi} = [0,1]^{k_{\xi}}$, $\mathcal{X}_s = [0,1]^2$ and $q = 3$. That is,

$$\mathbf{f}(\boldsymbol{\xi},\mathbf{x}_s) = ((p(\boldsymbol{\xi};\mathbf{x}_s), \mathbf{u}(\boldsymbol{\xi};\mathbf{x}_s)),$$

where $p(\boldsymbol{\xi};\mathbf{x}_s)$ and $\mathbf{u}(\boldsymbol{\xi};\mathbf{x}_s)$ is the solution of the boundary problem defined by Eqs. (4.30), (4.31) and (4.33) at the spatial point $\mathbf{x}_s$ for a permeability field given by Equation 4.35. Our purpose is to learn this map and also propagate the uncertainty of the stochastic variables through it by using a finite number of simulations.

The boundary value problem is solved using the Mixed Finite Element formulation. We use first-order Raviart-Thomas elements for the velocity [79], and zero-order discontinuous elements for the pressure [12]. The spatial domain is discretized using a 64×64 triangular mesh. The solver was implemented using the Dolfin C++ library [58]. The eigenfunctions of the exponential random field used to model the permeability were calculated via Stokhos which is part of Trilinos [49].

For each stochastic input $\boldsymbol{\xi}$, the response is observed on a regular 32×32 square spatial grid. Because of the regular nature of the spatial grid as well as the separable nature of the Square Exponential correlation function we use, it can be easily shown that the 1024×1024 spatial covariance matrix $\mathbf{A}_s$ can be written as

$$\mathbf{A}_s = \mathbf{A}_{s,1} \otimes \mathbf{A}_{s,2},$$

where $\mathbf{A}_{s,i}, i = 1, 2$ are 32×32 covariance matrices pertaining to the horizontal and vertical spatial directions, respectively. Of course, it is vital to make use of this simplification. The data collected this way are used to train a 3-dimensional Gaussian process which is then used to make predictions on the same spatial grid. We train our model, using in sequence 24, 64 and 120 observations of the deterministic solver in which the stochastic inputs are selected from a Latin hyper-cube design. The prior hyper-parameters are set to:

$$\begin{aligned}
\gamma_{\xi} &= 1/3 \\
\gamma_s &= 1/0.01 \\
\zeta_{\alpha} &= 10^2, \text{ for } \alpha = \boldsymbol{\xi}, s.
\end{aligned}$$

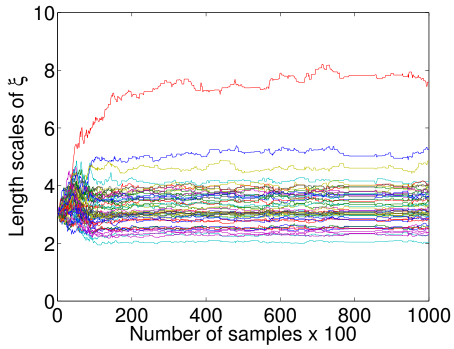
The initial values of the hyper-parameters used to start the Gibb's procedure are chosen to be the means of the priors. For each training set, we sample the posterior of the hyper-parameters $100,000$ times (see Figure 4.5 for a representative example). Then, we draw 100 sample surrogates as described in Algorithm 6. For each sampled surrogate, we calculate the statistics of interest. Finally, we compute and report the mean and the standard deviation of these statistics. The results are compared to Monte Carlo estimates.

Figure 4.6 compares the mean of the mean of u_x to a Monte Carlo estimate using 108,000 observations. Two standard deviations of the mean of u_x for the case of 120 observations are shown in subfigure (d). The same statistic for u_y and p is reported in Figs. 4.7 and 4.8, respectively. Figure 4.9 compares the mean of the variance of u_x to a Monte Carlo estimate using $108,000$ observations. Two standard deviations of the variance of u_x for the case of 120 observations are shown in subfigure (d). The same statistic for u_y and p is reported in Figs. 4.10 and 4.11, respectively. We observe - especially for the cases of 24 and 64 observations - that the variance is underestimated. Of course, this is to be expected given the very limited set of data available. Fortunately, the error bars seem to compensate for this under-estimation with the exception of the case that corresponds to the variance of the pressure p.

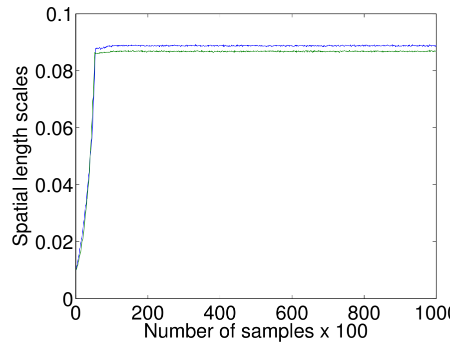
Figure 4.12 depicts the predicted probability densities of $u_x(0.5, 0.5)$ along with their error bars, for all available training sets. We see that the tails of the probability density are not estimated correctly. In particular, we observe two different types of potential problems. Firstly, the left hand side puts too much weight on negative values for $u_x(0.5, 0.5)$ even though it is quite clear (see subfigure (d)) that u_x is always positive on that particular spatial location. However, our prior assumption is that the response is a sample from a Gaussian random field. Hence, negative values for $u_x(0.5, 0.5)$ are very plausible. The model, can correct this belief only by observing an adequate number of data points. It is a fact, that all 120 observations of u_x near $(0.5, 0.5)$ are strictly positive. However, these observations are not enough to change the prior belief that $u_x(0.5, 0.5)$ might also take negative values. Notice, though, that as we go from (a), to (b), to (c), the trend is gradually corrected. If one wanted to incorporate the fact that a quantity of interest is strictly positive, then it is usually recommended to observe the logarithm of the quantity instead. Let us now get to the second problem which has to do with the underestimation of the right tail of the distribution. Let us start by noticing that cases (a) and (b) put enough weight on it. The reason for this is not that there are observations close to this region. It is again a consequence of the Gaussian assumption, just like in the first problem we discussed. However, for the case of 120 observations, we see that the model significantly underestimates the right tail. The reason, of course, is that there is not a single observation in the training set that takes values close to that region. One cannot possibly expect to capture a long tail without observing any events on it. The remedy here is a smarter choice of the observations on the lines of the active learning techniques that we have investigated in other places [**?**]. It is needless to say, that if the purpose of the practitioner is the investigation of improbable events, then she should favor active learning schemes that have a bias for extreme values. This is clearly beyond the scope of the present work. Finally, Figs. 4.13, 4.14 and 4.15 show the predicted probability densities of $u_x(0.25, 0.25)$, $p(0.5, 0.5)$ and $p(0.25, 0.25)$, respectively. The same comments as for the $u_x(0.5, 0.5)$ case are also applicable here. Finally, the joint probability density of $u_x(0.5, 0.5)$ and $u_y(0.5, 0.5)$ is shown in Fig. 4.16. We observe again, the underestimation of the top right long tail of the distribution and the broadening that occurs close to $(0, 0)$.

4.4 Conclusions

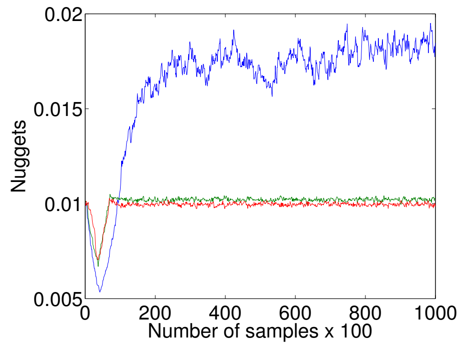
We developed a multi-output Gaussian process model that explicitly models the linear part of correlations between distinct outputs as well as space and/or time. By exploiting the static nature of the spatial/time inputs as well as the special



(a) Samples of $\mathbf{r}_\xi$.



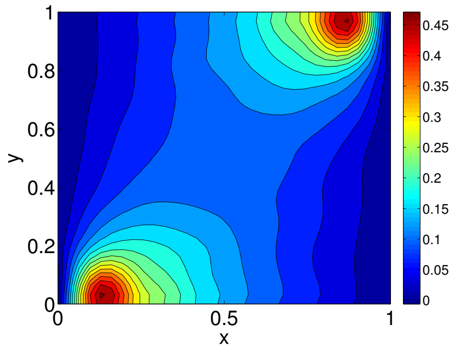
(b) Samples of $\mathbf{r}_s$.



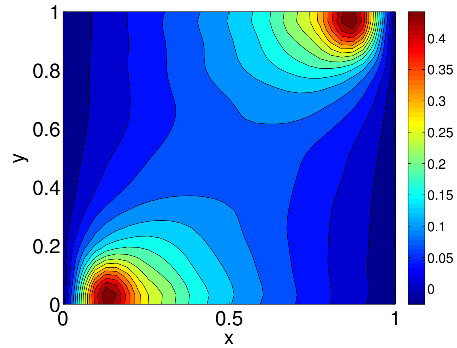
(c) Samples of g_ξ (blue), $g_{s,1}$ (green) and $g_{s,2}$ (red).

Figure 4.5: Porous flow: Samples drawn from the posterior of the hyper-parameters ((a) for r_ξ, (b) for $\mathbf{r}_s$ and (c) for the nuggets) for the case of 120 observations. It is apparent that the spatial length scales are clearly identified, while the hyper-parameters of the stochastic variables have a much wider posterior. Of course, this is expected given the limited number of observations.

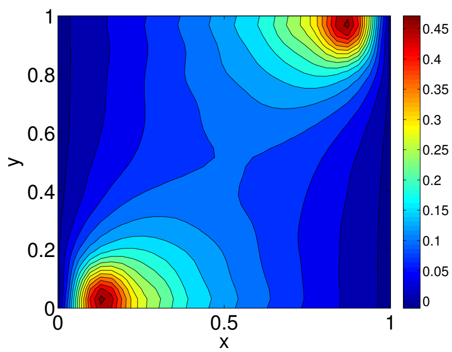
nature of separable covariance functions, we were able to express all required quantities for inference and predictions in terms of Kronecker products. This led to highly efficient computations both in terms of memory and CPU time. We recognized the fact that the posterior predictive distribution of the Gaussian process defines a probability measure on the function space of possible surrogates and we described an approximate method that yields kernel-based analytic sample surrogates. The scheme was applied to uncertainty quantification tasks in which we were able to quantify the epistemic uncertainty induced by



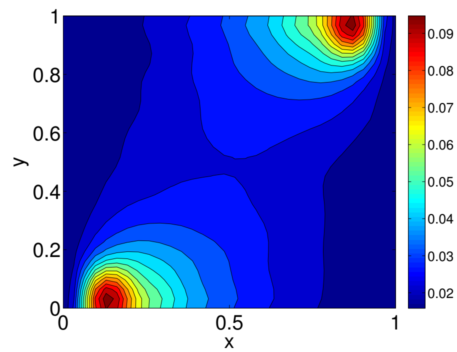
(a) Mean of the mean of u_x, 24 obs.



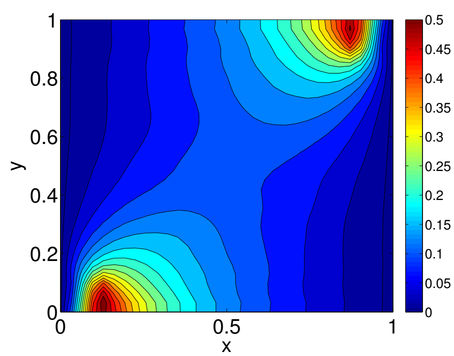
(b) Mean of the mean of u_x, 64 obs.



(c) Mean of the mean of u_x, 120 obs.

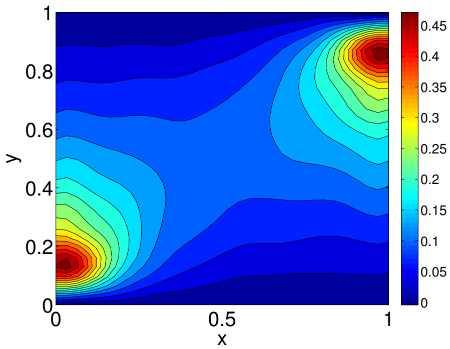


(d) Error bar of the mean of u_x, 120 obs.

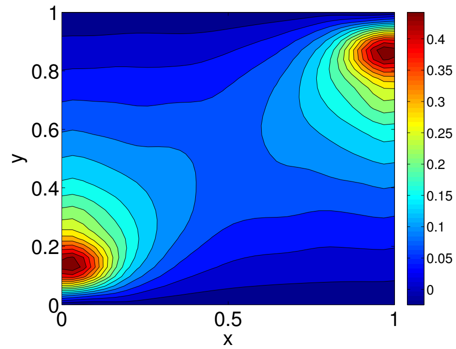


(e) MC estimate, 108,000 obs.

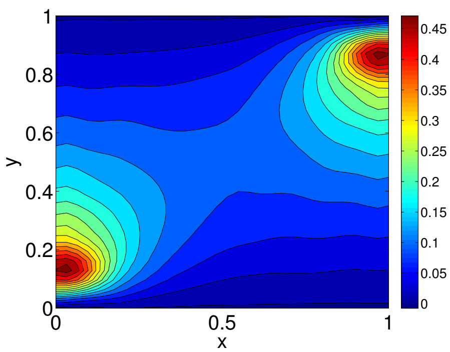
Figure 4.6: Porous flow: Mean of u_x. Subfigures (a), (b) and (c) show the mean of the mean of u_x for 24, 64 and 120 observations, respectively. Subfigure (d) plots two standard deviations of the mean of u_x for 120 observations. Finally, (e) shows the MC estimate of the same quantity using 108,000 observations.



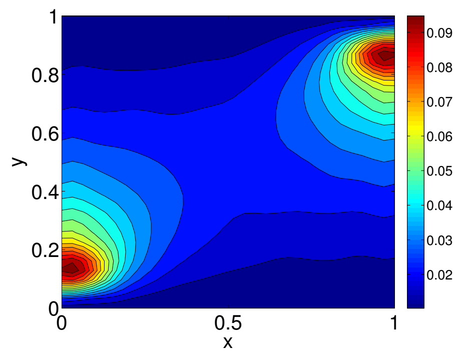
(a) Mean of the mean of u_y, 24 obs.



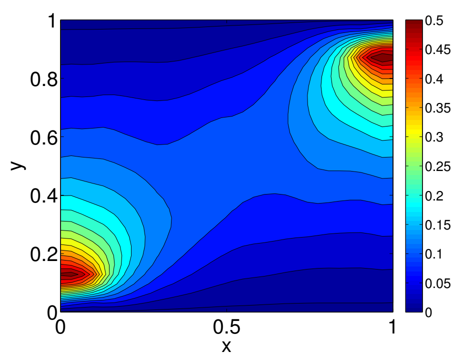
(b) Mean of the mean of u_y, 64 obs.



(c) Mean of the mean of u_y, 120 obs.

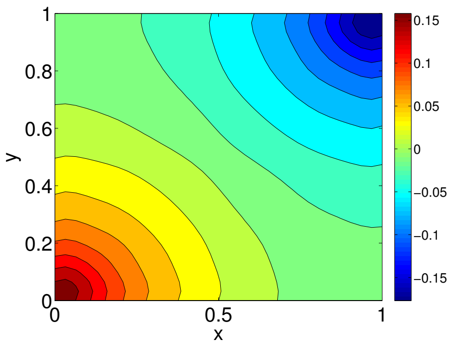


(d) Error bar of the mean of u_y, 120 obs.

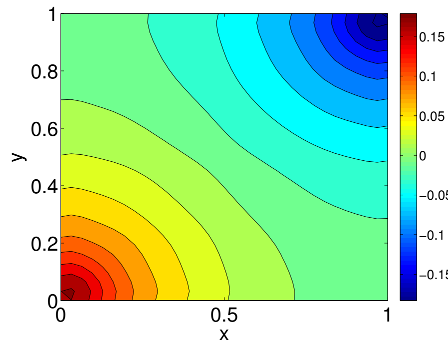


(e) MC estimate, 108,000 obs.

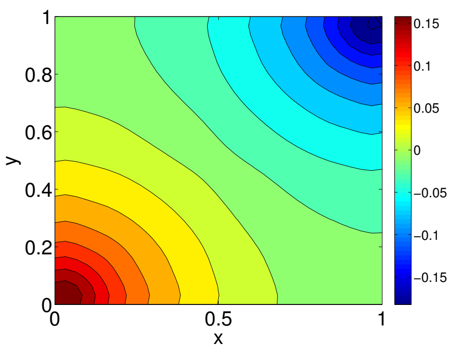
Figure 4.7: Porous flow: Mean of u_y. Subfigures (a), (b) and (c) show the mean of the mean of u_y for 24, 64 and 120 observations, respectively. Subfigure (d) plots two standard deviations of the mean of u_y for 120 observations. Finally, subfigure (e) shows the MC estimate of the same quantity using $108,000$ observations.



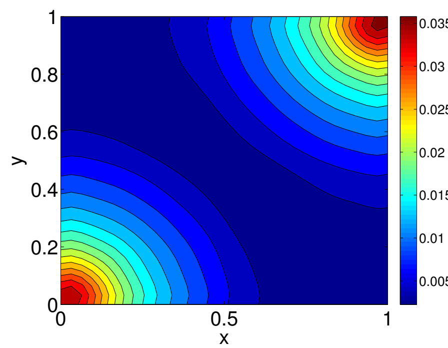
(a) Mean of the mean of p, 24 obs.



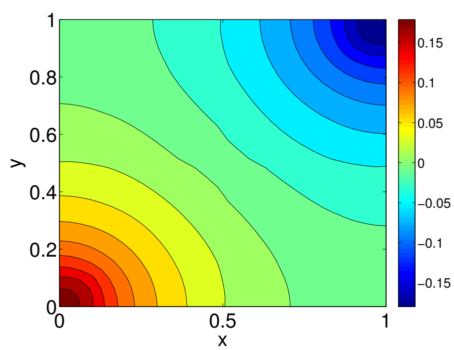
(b) Mean of the mean of p, 64 obs.



(c) Mean of the mean of p, 120 obs.

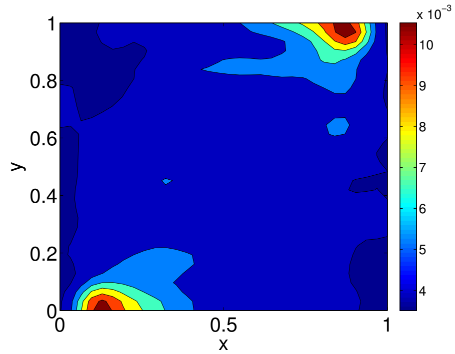


(d) Error bar of the mean of p, 120 obs.

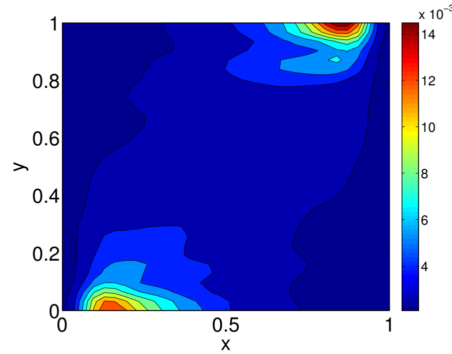


(e) MC estimate, 108,000 obs.

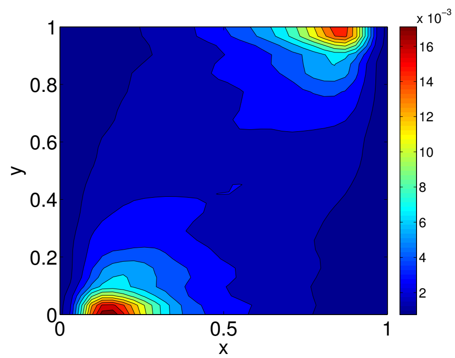
Figure 4.8: Porous flow: Mean of p. Subfigures (a), (b) and (c) show the mean of the mean of p for 24, 64 and 120 observations, respectively. Subfigure (d) plots two standard deviations of the mean of p for 120 observations. Finally, subfigure (e) shows the MC estimate of the same quantity using $108,000$ observations.



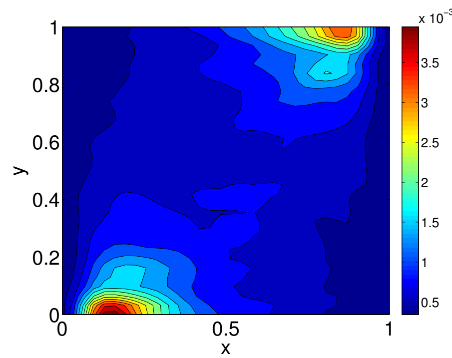
(a) Mean of the variance of u_x, 24 obs.



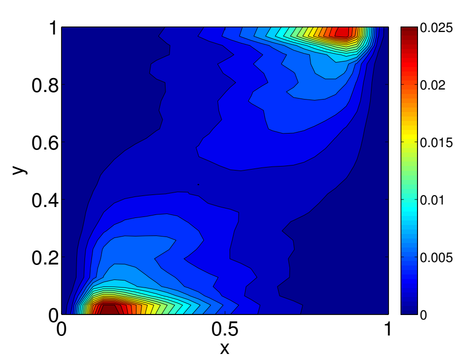
(b) Mean of the variance of u_x, 64 obs.



(c) Mean of the variance of u_x, 120 obs.

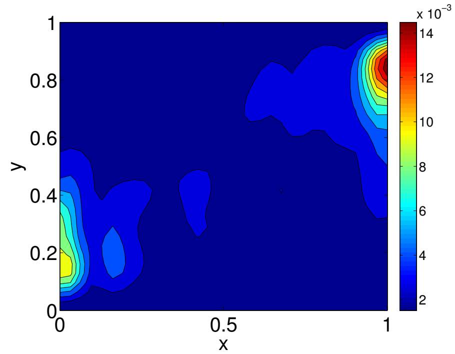


(d) Error bar of the variance of u_x, 120 obs.

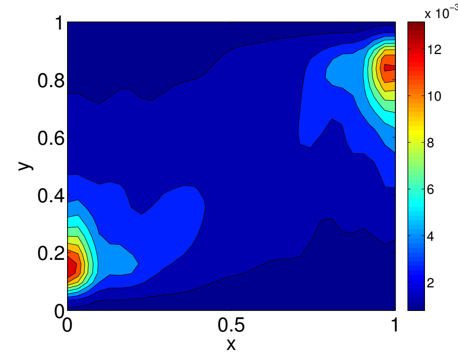


(e) MC estimate, 108,000 obs.

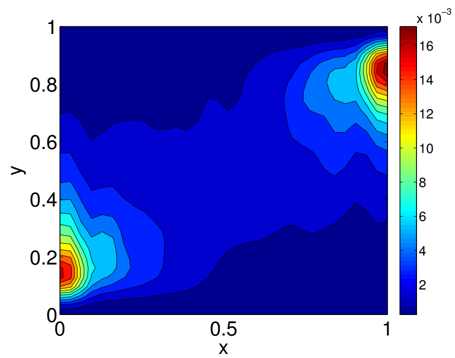
Figure 4.9: Porous flow: Variance of u_x. Subfigures (a), (b) and (c) show the mean of the variance of u_x for 24, 64 and 120 observations, respectively. Subfigure (d) plots two standard deviations of the variance of u_x for 120 observations. Finally, subfigure (e) shows the MC estimate of the same quantity using $108,000$ observations.



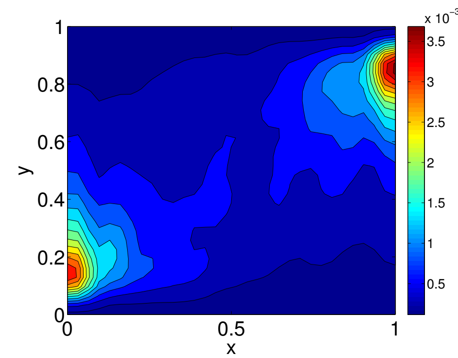
(a) Mean of the variance of u_y, 24 obs.



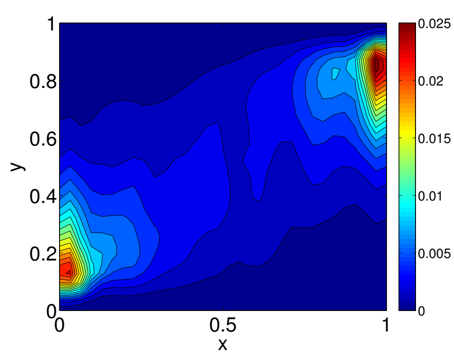
(b) Mean of the variance of u_y, 64 obs.



(c) Mean of the variance of u_y, 120 obs.

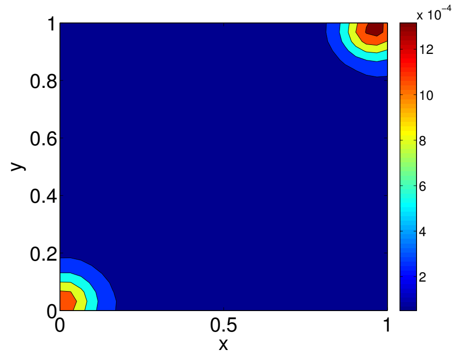


(d) Error bar of the variance of u_p, 120 obs.

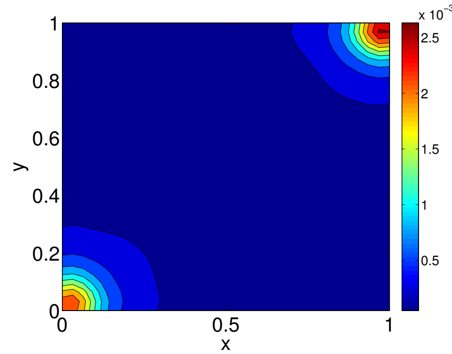


(e) MC estimate, 108,000 obs.

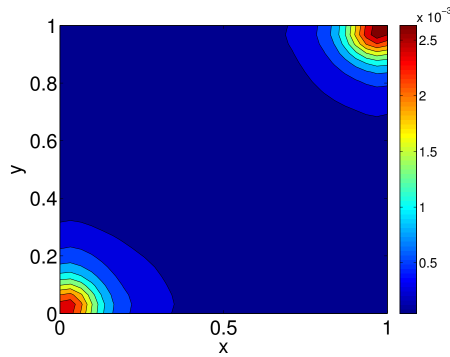
Figure 4.10: Porous flow: Variance of u_y. Subfigures (a), (b) and (c) show the mean of the variance of u_y for 24, 64 and 120 observations, respectively. Subfigure (d) plots two standard deviations of the variance of u_y for 120 observations. Finally, subfigure (e) shows the MC estimate of the same quantity using $108,000$ observations.



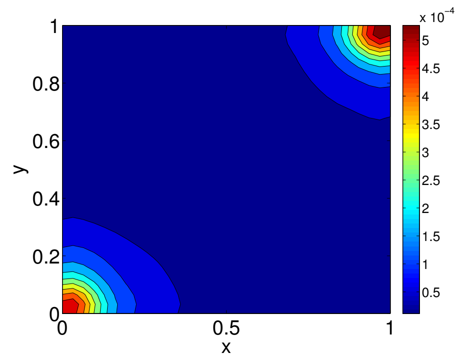
(a) Mean of the variance of p, 24 obs.



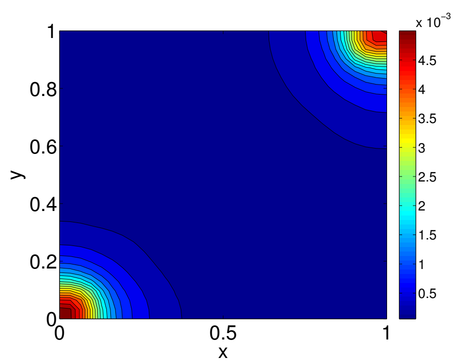
(b) Mean of the variance of p, 64 obs.



(c) Mean of the variance of p, 120 obs.

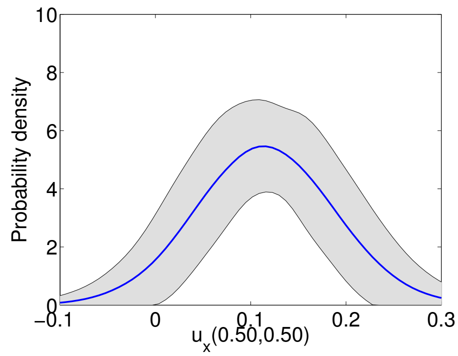


(d) Error bar of the variance of p, 120 obs.

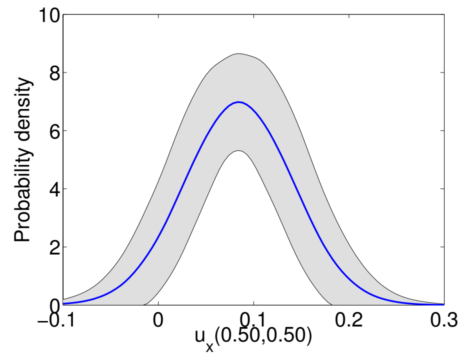


(e) MC estimate, 108,000 obs.

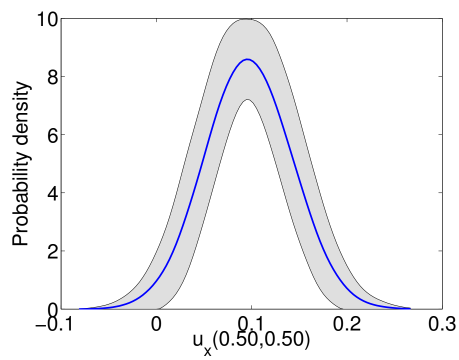
Figure 4.11: Porous flow: Variance of p. Subfigures (a), (b) and (c) show the mean of the variance of p for 24, 64 and 120 observations, respectively. Subfigure (d) plots two standard deviations of the variance of p for 120 observations. Finally, subfigure (e) shows the MC estimate of the same quantity using $108,000$ observations.



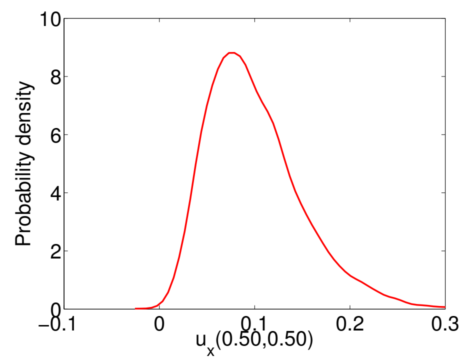
(a) 24 observations



(b) 64 observations



(c) 120 observations

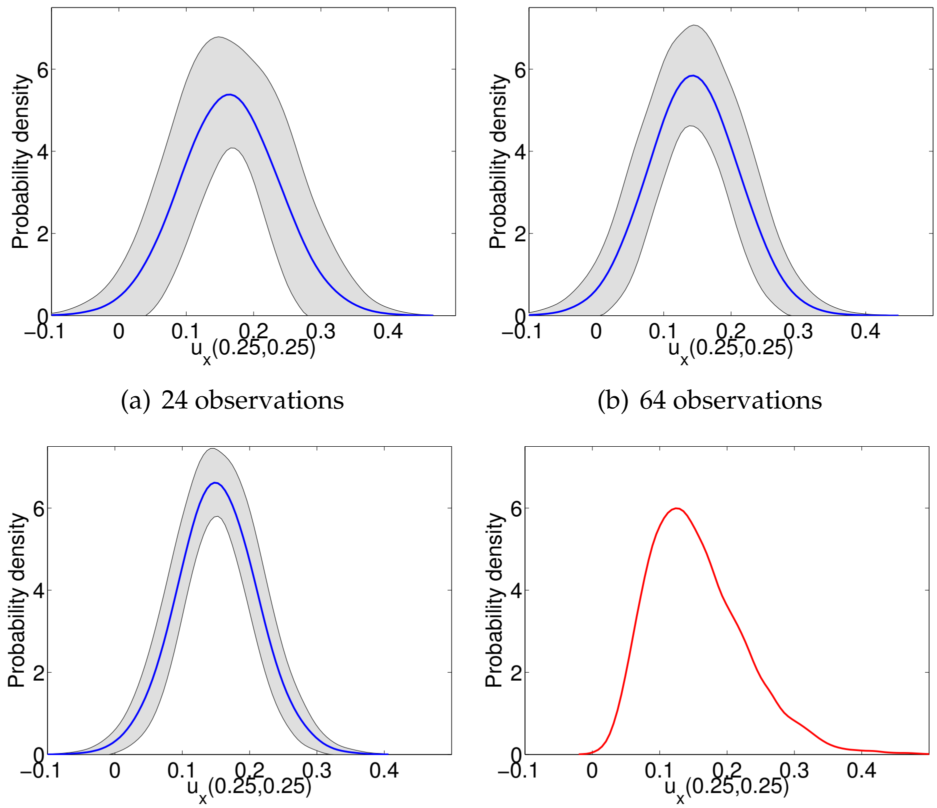


(d) MC, 10,000 samples

Figure 4.12: Porous flow: The PDF of $u_x(0.5, 0.5)$. The blue lines show the average PDF over 100 sampled surrogates for the cases of 24 (a), 64 (b) and 120 (c) observations. The filled gray area corresponds to two standard deviations of the PDFs about the mean PDF. The solid red line of (d) is the Monte Carlo estimate using $10,000$ observations.

the limited number of observations.

Despite the successes, we observe certain aspects that require further investigation. Firstly, we noticed a systematic underestimation of the tails of the predicted probability densities. Of course, this is expected in a limited observations setting. However, we are confident that the model can be considerably improved without losing in efficiency in several different ways. To start with, in the flow through porous media example, we can see that the assumption of sta-



(a) 24 observations

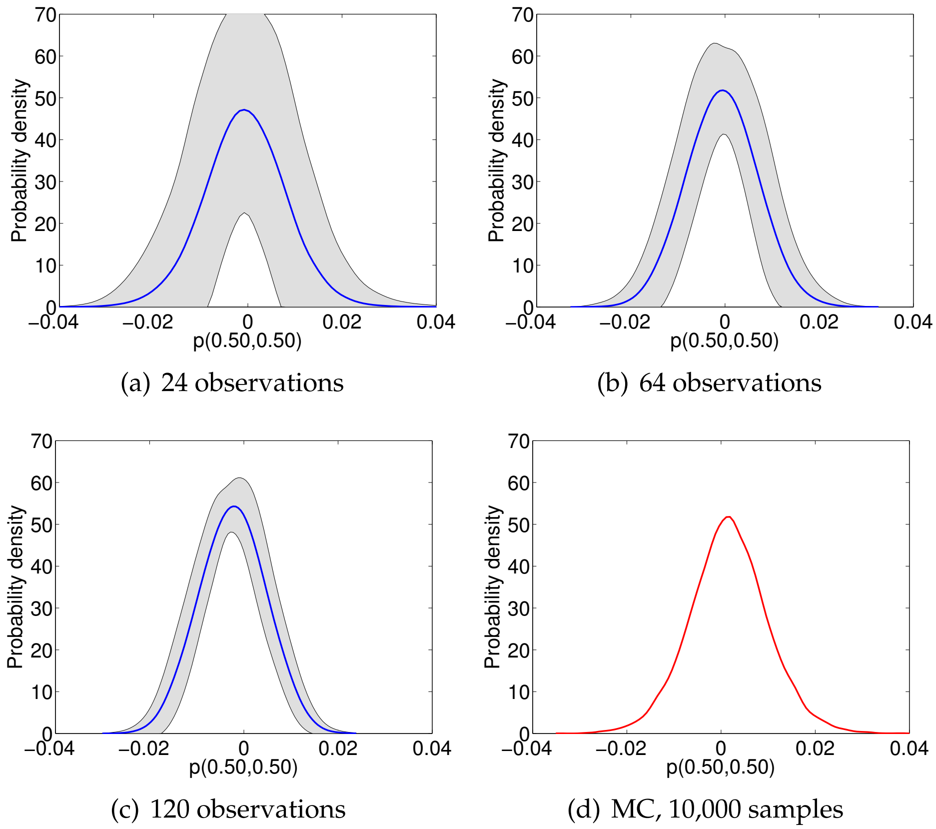
(b) 64 observations

(c) 120 observations

(d) MC, 10,000 samples

Figure 4.13: Porous flow: The pdf of $u_x(0.25, 0.25)$. The blue lines show the average PDF over 100 sampled surrogates for the cases of 24 (a), 64 (b) and 120 (c) observations. The filled gray area corresponds to two standard deviations of the PDFs about the mean PDF. The solid red line of (d) is the Monte Carlo estimate using $10,000$ observations.

tionarity in space is wrong. It is evident that the velocities vary more close to the wells, than they do away from them. The stationary covariance in space, forces the model to make a compromise in the spatial length scales. On one hand, regions close to the wells seem smoother than necessary while, on the other hand, regions away from them are more wavy. Hence, we are expecting that using a non-stationary covariance or a tree-based model will improve the situation significantly [**?**]. Furthermore, it would be very interesting to see how the results would change if a sequential active learning approach was followed for the col-



(a) 24 observations

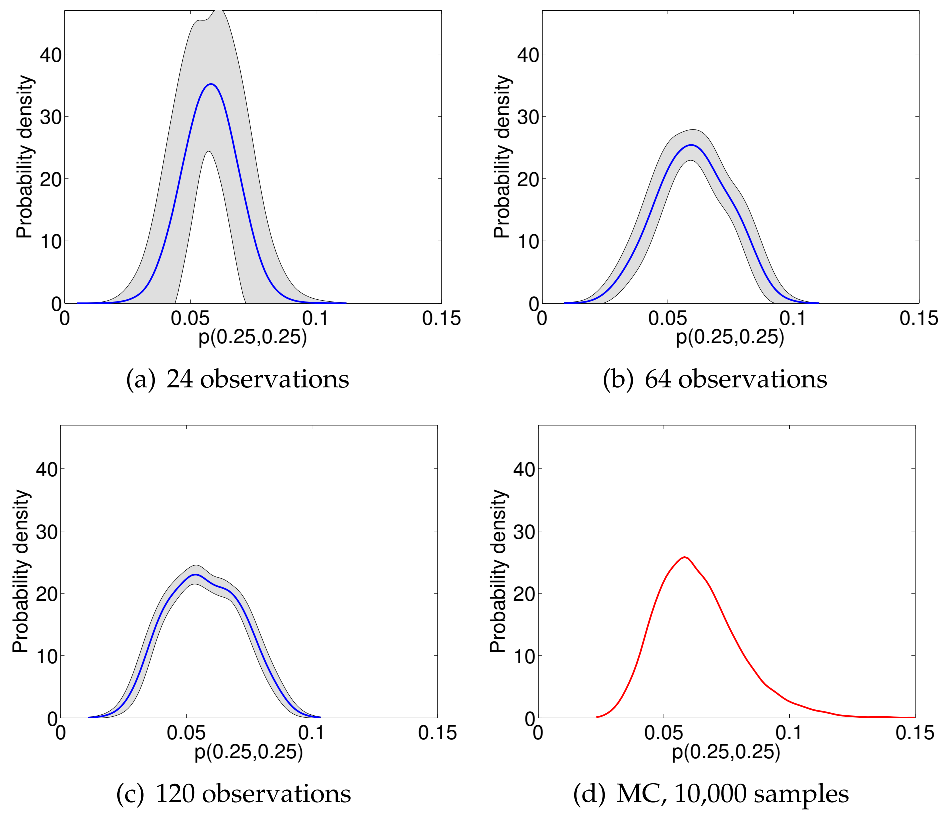
(b) 64 observations

(c) 120 observations

(d) MC, 10,000 samples

Figure 4.14: Porous flow: The pdf of $p(0.5, 0.5)$. The blue lines show the average PDF over 100 sampled surrogates for the cases of 24 (a), 64 (b) and 120 (c) observations. The filled gray area corresponds to two standard deviations of the PDFs about the mean PDF. The solid red line of (d) is the Monte Carlo estimate using $10,000$ observations.

lection of the observations. It seems plausible, that the most effective way to improve the tails of the distributions would be to select observations with extreme properties. A simple variance-based active learning scheme seems inadequate (of course here we are talking about the case in which the observations are kept to very small number). The particulars of an alternative are a very interesting research topic.



(a) 24 observations

(b) 64 observations

(c) 120 observations

(d) MC, 10,000 samples

Figure 4.15: Porous flow: The PDF of $p(0.25, 0.25)$. The blue lines show the average PDF over 100 sampled surrogates for the cases of 24 (a), 64 (b) and 120 (c) observations. The filled gray area corresponds to two standard deviations of the PDFs about the mean PDF. The solid red line of (d) is the Monte Carlo estimate using $10,000$ observations.

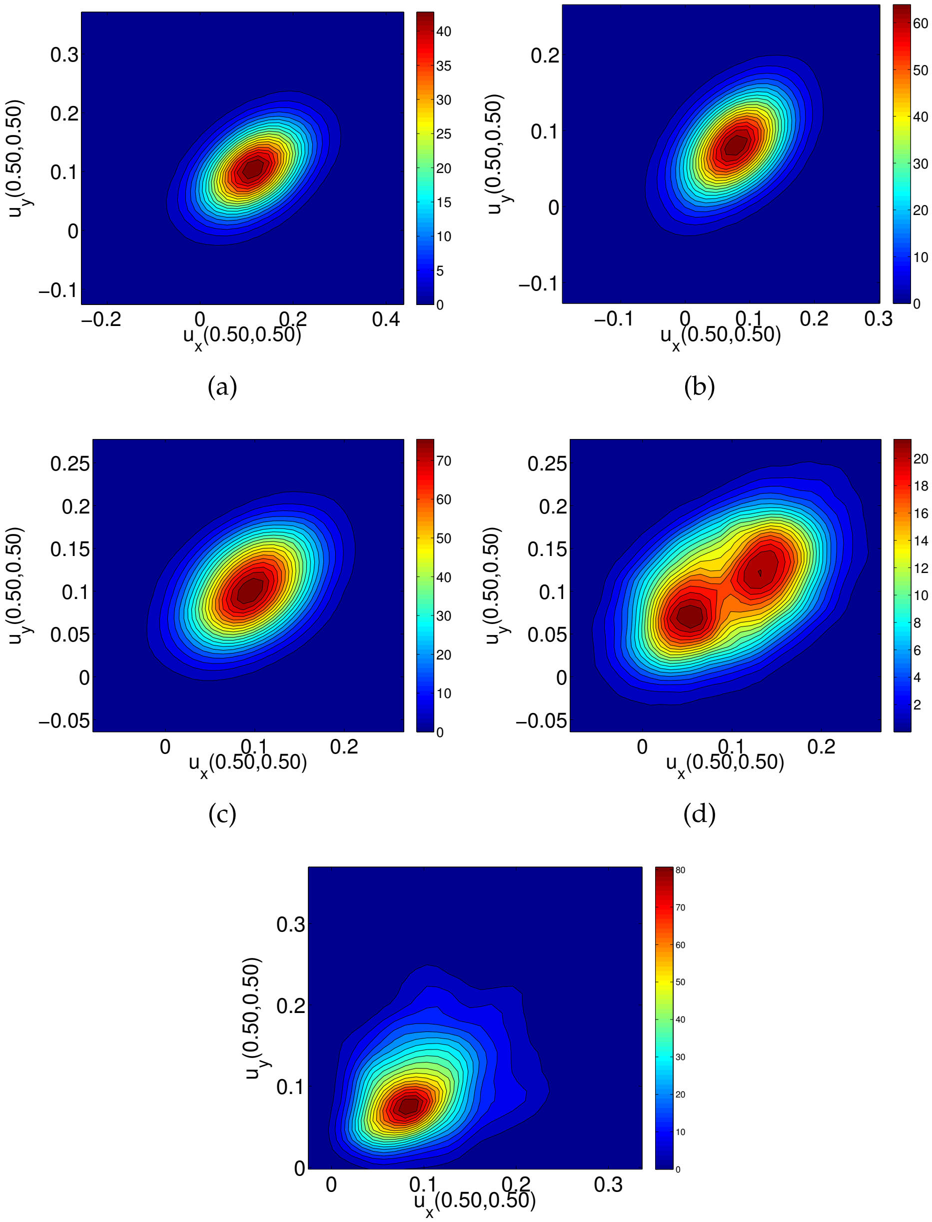


Figure 4.16: Porous flow: The joint PDF of $u_x(0.5, 0.5)$ and $u_y(0.5, 0.5)$. The contours show the average joint PDF over 100 sampled surrogates for the cases of 24 (a), 64 (b) and 120 (c) observations, respectively. Subfigure (d) plots two standard deviations of the joint PDF for 120 observations. Finally, subfigure (e) shows the MC estimate of the same quantity using 10,000 observations.

APPENDIX A

MATHEMATICAL DETAILS FOR RELEVANT VECTOR MACHINES

A.1 Splitting the evidence

What follows is basically a generalization to the multi-output case of the ideas found in [89]. By making repeated use of the matrix determinant lemma [46] and the Woodbury matrix identity [44], it is possible to show that:

$$\begin{aligned}\mathcal{E}(\boldsymbol{\alpha}|\mathcal{D}_{\mathrm{sc}}) &= -\frac{1}{2}\log 2\pi \\ &\quad -\frac{1}{2N}\left(\log|\mathbf{C}_{-s}| - \log\alpha_s + \log|\alpha_s + \phi_s^T\mathbf{C}_{-s}^{-1}\boldsymbol{\phi}_s|\right) \\ &\quad -\frac{1}{2MN}\sum_{r=1}^{M}\mathbf{z}_r^T\left(\mathbf{C}_{-s}^{-1} - \frac{\mathbf{C}_{-s}^{-1}\boldsymbol{\phi}_s\boldsymbol{\phi}_s^T\mathbf{C}_{-s}^{-1}}{\alpha_s + \boldsymbol{\phi}_s^T\mathbf{C}_{-s}^{-1}\boldsymbol{\phi}_s}\right)\mathbf{z}_r \\ &= \mathcal{E}(\boldsymbol{\alpha}_{-s}) + \epsilon(\alpha_s),\end{aligned}$$

where $\mathcal{E}(\boldsymbol{\alpha}_{-s})$ is the evidence with α_s removed and $\epsilon(\alpha_s)$ is given by:

$$\epsilon(\alpha_s) = \frac{1}{2N}\log\alpha_s - \frac{1}{2N}\log|\alpha_s + h_s| + \frac{1}{2MN}\frac{\sum_{r=1}^{M} q_{rs}^2}{\alpha_s + h_s}, \tag{A.1}$$

where we have introduced the intermediate statistics:

$$h_s = \boldsymbol{\phi}_s^T\mathbf{C}_{-s}^{-1}\boldsymbol{\phi}_s \text{ and } q_{rs} = \boldsymbol{\phi}_s^T\mathbf{C}_{-s}^{-1}\mathbf{z}_r. \tag{A.2}$$

In practice, it is more convenient to keep track of the following statistics:

$$H_s = \boldsymbol{\phi}_s^T\mathbf{C}^{-1}\boldsymbol{\phi}_s \text{ and } Q_{rs} = \boldsymbol{\phi}_s^T\mathbf{C}^{-1}\mathbf{z}_r, \tag{A.3}$$

since:

$$h_s = \frac{\alpha_s H_s}{\alpha_s - H_s}, \text{ and } q_{rs} = \frac{\alpha_s Q_{rs}}{\alpha_s - H_s}. \tag{A.4}$$

A.2 Stationary points of $\epsilon(\alpha_s)$

The derivative of $\epsilon(\alpha_s)$ is:

$$\frac{\partial \epsilon(\alpha_s)}{\partial \alpha_s} = \frac{h_s^2 - \alpha_s \left(\frac{1}{M}\sum_{r=1}^{M} q_{rs}^2 - h_s\right)}{2\alpha_s N(\alpha_s + h_s)^2} = \frac{h_s^2 - \alpha_s \theta_s}{2\alpha_s N(\alpha_s + h_s)^2}, \tag{A.5}$$

where

$$\theta_s = \frac{1}{M}\sum_{r=1}^{M} q_{rs}^2 - h_s. \tag{A.6}$$

We are basically interested in maximizing $\epsilon(\alpha_s)$ with respect to $\alpha_s > 0$. We observe that there exist two possible cases:

1. If $\theta_s > 0$, then $\epsilon(\alpha_s)$ has a unique stationary point at

$$\alpha_s = \frac{h_s^2}{\theta_s}. \tag{A.7}$$

   By evaluating the second derivative of $\epsilon(\alpha_s)$ at $\alpha_s = h_s^2/\theta_s$, it is straightforward to check that this is indeed the maximum of the function.

2. If $\theta_s \leq 0$, then $\epsilon(\alpha_s)$ is maximized at

$$\alpha_s = +\infty, \tag{A.8}$$

   since $\partial\epsilon/\partial\alpha_s$ is positive for all $\alpha_s > 0$. This effectively removes the s-th basis function from the model, since its corresponding weight is identically set to zero.

A.3 Possible actions

Whenever some $\alpha_s = +\infty$, the corresponding basis functions can be thought of as being out of the model. Assume that currently there are $1 \leq S_{\text{in}} \leq S$ basis functions in the model. The design matrix $\mathbf{\Phi}$ is built only upon the basis functions

$\boldsymbol{\phi}_s$ for which $\alpha_s < \infty$, i.e. it is an $N \times S_{\text{in}}$ matrix. The mean of the weights $\boldsymbol{\mu}_r$ and their covariance matrix $\boldsymbol{\Sigma}$ defined in Equation 3.18 are an S_{in}-dimensional vector and an $S_{\text{in}} \times S_{\text{in}}$ matrix, respectively. Also, assume that the statistics H_s, Q_{rs} given in Eq. (A.3) are already calculated for all $s = 1, \ldots, S$. At each step of the algorithm we update a single α_s. There are three possible actions: 1) *add* a new basis function to the model; 2) *re-estimate* the hyper-parameter of an existing basis function and 3) *remove* a basis function from the model. The action that results in the maximum change in the evidence $\Delta\mathcal{E}^{\text{action}} = \mathcal{E}(\boldsymbol{\alpha}^{\text{new}}) - \mathcal{E}(\boldsymbol{\alpha})$ is selected. For completeness, we list how the possible actions can be distinguished, how the change in evidence can be calculated and how the various statistics can be updated iteratively:

1. If $\theta_s > 0$ and $\alpha_s = \infty$, then $\boldsymbol{\phi}_s$ is a candidate for addition. The value for α_s maximizing the evidence is:

$$\alpha_s^{\text{new}} = \frac{h_s^2}{\theta_s},$$

   yielding a change in evidence:

$$\Delta\mathcal{E}^{\text{add}} = \epsilon(\alpha_s^{\text{new}}).$$

2. If $\theta_s > 0$ and $\alpha_s < \infty$, then $\boldsymbol{\phi}_s$ is a candidate for re-estimation. The value for α_s maximizing the evidence is:

$$\alpha_s^{\text{new}} = \frac{h_s^2}{\theta_s},$$

   yielding a change in evidence:

$$\Delta\mathcal{E}^{\text{re-estimate}} = \epsilon(\alpha_s^{\text{new}}) - \epsilon(\alpha_s).$$

3. If $\theta_s \leq 0$ and $\alpha_s < \infty$, then $\boldsymbol{\phi}_s$ is a candidate for removal. The value for α_s maximizing the evidence is:

$$\alpha_s^{\text{new}} = \infty,$$

yielding a change in evidence:

$$\Delta\mathcal{E}^{\text{re-estimate}} = -\epsilon(\alpha_s).$$

A.4 Implementation details

Let us consider again the matrix $\mathbf{C}$ defined in Equation 3.24. As before, let $\mathbf{\Phi}$ be the design matrix of all relevant basis functions (the ones for which $\alpha_s < \infty$). These are the only ones contributing to $\mathbf{C}$ and without loss of generality we may assume that the rest are not present in the model. Using the Woodbury matrix identity [44], we can show that:

$$\mathbf{C}^{-1} = \beta\mathbf{I} - \beta\mathbf{\Phi}\left(\beta^{-1}\operatorname{diag}(\boldsymbol{\alpha}) + \mathbf{\Phi}^T\mathbf{\Phi}\right)^{-1}\mathbf{\Phi}^T,$$

where, of course $\boldsymbol{\alpha}$ is the vector of only the finite hyper-parameters. Define the $S_{\text{in}} \times S_{\text{in}}$ matrix $\mathbf{A}$ by:

$$\mathbf{A} = \beta^{-1/2}\operatorname{diag}(\boldsymbol{\alpha})^{1/2},$$

and notice that the part that needs to be inverted has the familiar form $\mathbf{A}^T\mathbf{A} + \mathbf{\Phi}^T\mathbf{\Phi}$ found in regularized least squares problems. This suggests using the Generalized Singular Value Decomposition (GSVD) of the pair $(\mathbf{A}, \mathbf{\Phi})$ in order to carry out the computations. The GSVD is given by:

$$\mathbf{\Phi} = \mathbf{U}\mathbf{\Sigma}_1[0\ \mathbf{R}]\mathbf{Q}^T \text{ and } \mathbf{A} = \mathbf{V}\mathbf{\Sigma}_2[0\ \mathbf{R}]\mathbf{Q}^T,$$

where $\mathbf{U} \in \mathbb{R}^{N\times N}, \mathbf{V} \in \mathbb{R}^{S_{\text{in}}\times S_{\text{in}}}$ and $\mathbf{Q} \in \mathbb{R}^{S_{\text{in}}\times S_{\text{in}}}$ are orthogonal, $\mathbf{\Sigma}_1 \in \mathbb{R}^{N\times r}$ and $\mathbf{\Sigma}_2 \in \mathbb{R}^{S_{\text{in}}\times r}$ are zero except for the diagonal and have the property:

$$\mathbf{\Sigma}_1^T\mathbf{\Sigma}_1 + \mathbf{\Sigma}_2^T\mathbf{\Sigma}_2 = I,$$

$\mathbf{R} \in \mathbb{R}^{r\times r}$ is upper triangular and non-singular, while r is the rank of $[\mathbf{\Phi}^T \;\; \mathbf{A}^T]$. When the number of samples is greater or equal than the relevant basis functions, i.e. $N \geq S_{\text{in}}$, then $r = S_{\text{in}}$ since $\mathbf{A}$ is non-singular. We will always assume that this is the case, since it does not make sense to include in the model more basis functions than observations. We want to make clear at this point the distinction between the pool of basis functions from which we choose and the relevant basis functions. The former can be as large as we want but the latter should be less or equal to the number of available observations. In this case, the matrix of zeros disappears and we obtain:

$$\mathbf{\Phi} = \mathbf{U}\mathbf{\Sigma}_1\mathbf{R}\mathbf{Q}^T \text{ and } \mathbf{A} = \mathbf{V}\mathbf{\Sigma}_2\mathbf{R}\mathbf{Q}^T.$$

We can now notice that

$$\mathbf{A}^T\mathbf{A} + \mathbf{\Phi}^T\mathbf{\Phi} = \mathbf{Q}\mathbf{R}^T\mathbf{R}\mathbf{Q}^T,$$

and as a result

$$\mathbf{C}^{-1} = \beta\left(\mathbf{I} - \left(\mathbf{\Sigma}_1^T\mathbf{U}^T\right)^T\left(\mathbf{\Sigma}_1^T\mathbf{U}^T\right)\right).$$

The H_s statistic now takes the form

$$H_s = \beta\left(\| \boldsymbol{\phi}_s \|_2^2 - \| \mathbf{\Sigma}_1^T\mathbf{U}^T\boldsymbol{\phi}_s \|_2^2\right). \tag{A.9}$$

Similarly, the Q_{rs} statistic becomes

$$Q_{rs} = \beta\left(\phi_s^T\mathbf{z}_r - \left(\mathbf{\Sigma}_1^T\mathbf{U}^T\boldsymbol{\phi}_s\right)^T\left(\mathbf{\Sigma}_1^T\mathbf{U}^T\mathbf{z}_r\right)\right), \tag{A.10}$$

while the $\mathbf{z}_r^T\mathbf{C}^{-1}\mathbf{z}_r$ part of the evidence becomes:

$$\mathbf{z}_r^T\mathbf{C}^{-1}\mathbf{z}_r = \beta\left(\| \mathbf{z}_r \|_2^2 - \| \mathbf{\Sigma}_1^T\mathbf{U}^T\mathbf{z}_r \|_2^2\right). \tag{A.11}$$

The only part that remains in order to finalize the computation of the likelihood is $|\mathbf{C}|$. For this, we use the matrix-determinant lemma to obtain:

$$\begin{aligned}
|\mathbf{C}| &= |\beta^{-1}\mathbf{I} + \mathbf{\Phi} \operatorname{diag} \boldsymbol{\alpha}^{-1} \mathbf{\Phi}^T| && \text{(A.12)} \\
&= |\operatorname{diag} \boldsymbol{\alpha}^{-1}| \cdot |\beta^{-1} \operatorname{diag} \boldsymbol{\alpha} + \mathbf{\Phi}^T \mathbf{\Phi}| \beta^{S_{\text{in}}-N} && \text{(A.13)} \\
&= |\operatorname{diag} \boldsymbol{\alpha}^{-1}| \cdot |\mathbf{Q}\mathbf{R}^T \mathbf{R}\mathbf{Q}| \beta^{S_{\text{in}}-N} && \text{(A.14)} \\
&= |\operatorname{diag} \boldsymbol{\alpha}^{-1}| \cdot |\mathbf{R}|^2 \beta^{S_{\text{in}}-N}, && \text{(A.15)}
\end{aligned}$$

since $\mathbf{Q}$ is orthogonal. This gives us:

$$\log|\mathbf{C}| = -\sum_{i=1}^{S_{\text{in}}} \log \alpha_i + 2\sum_{i=1}^{S_{\text{in}}} \log|R_{ii}| + (S_{\text{in}} - N)\log\beta. \qquad \text{(A.16)}$$

Finally, predictions can be carried out easily by noticing that

$$\mathbf{\Sigma}^{-1} = \beta\left(\mathbf{A}^T\mathbf{A} + \mathbf{\Phi}^T\mathbf{\Phi}\right) = \beta\mathbf{Q}\mathbf{R}^T\mathbf{R}\mathbf{Q}^T.$$

For example, the mean of the weights can be found by solving the system:

$$(\mathbf{R}\mathbf{Q}^T)\boldsymbol{\mu}_r = \mathbf{\Sigma}_1^T \mathbf{U}^T \mathbf{z}_r,$$

which is a simple operation since $\mathbf{R}$ is upper triangular and $\mathbf{Q}$ orthogonal.

APPENDIX B

IMPLEMENTATION DETAILS FOR MULTI-OUTPUT GAUSSIAN PROCESS REGRESSION

In this appendix, we discuss several details with regards to the implementation of the UQ framework presented.

The nugget The covariance function we use has the special form:

$$c(\mathbf{x}^{(n)}, \mathbf{x}^{(m)}; \boldsymbol{\theta}, g) = c(\mathbf{x}^{(n)}, \mathbf{x}^{(m)}; \boldsymbol{\theta}) + g^2\delta_{nm}, \tag{B.1}$$

where $c(\cdot, \cdot; \boldsymbol{\theta})$ is a normal covariance function depending on some hyper-parameters $\boldsymbol{\theta}$, $g^2 > 0$ and δ_{nm} is the Kronecker delta. Such a covariance function corresponds to the case where $\mathbf{f}(\mathbf{x})$ is observed with additive Gaussian noise with zero mean and variance g^2 (see p. 16 of [77]). In the literature of analysis of computer experiments using GPs, g^2 is known as the *nugget*. Many authors (e.g. [81], [82]), omit the nugget on the grounds that computer codes are deterministic. Inclusion of the nugget, however, has been observed to enhance numerical stability in factorizing the covariance matrix [68, 41]. On our part, we have observed that numerical stability is further improved, if a zero mean, g^2 Gaussian noise is added to the scaled observed responses Equation 2.11. The effect of the nugget is the addition of a g^2 term in the predictive variance of the scaled responses. A typical value of the nugget we use in the numerical examples is $g^2 = 10^{-6}$. For a very recent discussion on the importance of the nugget in computer modeling, see [43].

Maximizing the marginal likelihood In this work, we make exclusive use of the SE covariance function defined in Equation 2.52. Its hyper-parameters are the signal strength $s_f > 0$ and the length scale of each stochastic input $\ell_k > 0$. Each stochastic element is associated with its local hyper-parameters which are found by maximizing the joint marginal likelihood subject to the positivity constraint. In order to achieve this in practice, we maximize with respect to the logarithm of these quantities, i.e. we re-parameterize the covariance function as:

$$\theta_1 = \log s_f,\ \theta_{k+1} = \log \ell_k.$$

This results in an equivalent unconstrained optimization problem which we solve using a Conjugate Gradient (CG) method [76], i.e. Equation 2.15 with $\mathbf{\Theta} = \mathbb{R}^{K+1}$. It is important to notice that the nugget, g^2, is not optimized. It remains fixed to a given small value. Specifically, we used the Fletcher-Reeves CG algorithm [27] as implemented in GSL [30]. The starting values $\boldsymbol{\theta}_0 = (\theta_{1,0}, \dots, \theta_{K+1,0})$ of the optimization algorithm are chosen as follows:

1. If we fit a GP for the first time (i.e. using $\mathbf{X}$ itself as the first element), we set $\theta_{1,0} = 0$ for the signal parameter and

   $$\theta_{k+1,0} = \log\left(\frac{1}{3}L_k\right),\ k = 1, \dots, K,$$

   for the length scale parameters, where $L_k = b_k - a_k$ is the extent of $\mathbf{X}$ along the k-dimension (Equation 3.39).

2. Otherwise, if $\mathbf{X}^i$ comes from splitting in half a parent element, we set $\boldsymbol{\theta}_0$ equal to the hyper-parameters of the parent element.

The optimization problem does not necessarily have a unique maximum. In reality, different local maxima are associated with different interpretations of the observed data set (Ch. 5 of [77]). In our numerical examples, we did not encounter any problems with this optimization and the maxima we obtained were quite robust. Powers of the response function are also treated as MGPs with SE covariance function, albeit having their own hyper-parameters $\boldsymbol{\theta}^q$ (see Section 2.2.2). These are also selected by maximizing the marginal likelihood.

Evaluation of the integrals Finally, we come to the problem of computing the necessary integrals for the evaluation of the statistics (Eqs. (2.27), (2.28) and (2.29)). It is apparent that for general elements, input probability distribution and covariance function, these integrals have to be numerically evaluated. We choose to work with square elements, uniform input probability distribution and SE covariance function. With this choice, it is possible to express those integrals analytically using the error function:

$$\Phi(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt. \tag{B.2}$$

In particular, let $\mathbf{X}^i = \times_{k=1}^K [a_k^i, b_k^i]$ and $p^i(\mathbf{z})$ be the uniform distribution on $\mathbf{X}^i$, i.e.

$$p^i(\mathbf{z}) = \frac{1_{\mathbf{X}^i}(\mathbf{z})}{\prod_{k=1}^K (b_k^i - a_k^i)}. \tag{B.3}$$

Then, it is easy to show that (for $q = 1$):

$$\epsilon^1(\mathbf{x}) = s_f^2 \left(\frac{\pi}{2}\right)^{K/2} \prod_{k=1}^K \ell_k^i \left(\Phi\left(\frac{b_k^i - x_k}{\sqrt{2}\ell_k^i}\right) - \Phi\left(\frac{a_k^i - x_k}{\sqrt{2}\ell_k^i}\right)\right) \tag{B.4}$$

and

$$\nu^1(\mathbf{x}, \mathbf{y}) = \left(\frac{\pi}{2}\right)^{K/2} s_f^3 \sqrt{c(\mathbf{x}, \mathbf{y})} \prod_{k=1}^K \ell_k^i \left(\Phi\left(\frac{2b_k^i - x_k - y_k}{2\ell_k^i}\right) - \Phi\left(\frac{2a_k^i - x_k - y_k}{2\ell_k^i}\right)\right). \tag{B.5}$$

The constant c^1 (Equation 2.28), can be trivially shown to be

$$c^1 = s_f^2. \tag{B.6}$$

The integrals that pertain to the higher moments $q > 1$ are obtained similarly by replacing the hyper-parameters with the ones that correspond to the MGP representing the response raised to the q power $\mathbf{f}^q$.

APPENDIX C

MATHEMATICAL DETAILS FOR MULTI-OUTPUT GAUSSIAN PROCESS REGRESSION

C.1 Kronecker Product Properties

C.1.1 Calculating matrix-vector and matrix-matrix products

Matrix-vector product Let $\mathbf{A} \in \mathbb{R}^{m_1 \times n_1}, \mathbf{B} \in \mathbb{R}^{m_2 \times n_2}$ and $\mathbf{x} \in \mathbb{R}^{n_1 n_2}$. We wish to calculate:

$$\mathbf{y} = (\mathbf{A} \otimes \mathbf{B})\, \mathbf{x} \in \mathbb{R}^{m_1 m_2},$$

without explicitly forming the Kronecker product. This may be easily achieved by exploiting the properties of the vectorization operation $\mathrm{vec}(\cdot)$ [63]. Let $\mathbf{X} \in \mathbf{R}^{n_2 \times n_1}$ be the matrix formed by folding the vector $\mathbf{x}$ so that $\mathbf{x} = \mathrm{vec}(\mathbf{X})$. Then we obtain:

$$\mathbf{y} = \mathrm{vec}(\mathbf{B}\mathbf{X}\mathbf{A}^T). \tag{C.1}$$

So all we need to do is two matrix multiplications. Notice that for the case of triangular $\mathbf{A}$ and $\mathbf{B}$ no additional memory is required.

Matrix-matrix product Let $\mathbf{A}$ and $\mathbf{B}$ be as before and $\mathbf{X} \in \mathbb{R}^{(n_1 n_2) \times s}$. We wish to calculate:

$$\mathbf{Y} = (\mathbf{A} \otimes \mathbf{B})\, \mathbf{X} \in \mathbb{R}^{(m_1 m_2) \times s}.$$

This can be trivially calculated by working column by column using the results of the previous subsection.

Three Kronecker products Let $\mathbf{C} \in \mathbb{R}^{m_3 \times n_3}$ and $\mathbf{x} \in \mathbb{R}^{n_1 n_2 n_3}$. Then the product

$$\mathbf{y} = (\mathbf{A} \otimes \mathbf{B} \otimes \mathbf{C})\,\mathbf{x} \in \mathbb{R}^{m_1 m_2 m_3},$$

can by calculated by observing that:

$$\mathbf{y} = \mathrm{vec}\left(\mathbf{C}\mathbf{X}\,(\mathbf{A} \otimes \mathbf{C})^T\right),$$

where $\mathbf{X} \in \mathbb{R}^{n_3 \times (n_1 n_2)}$ such that $\mathbf{x} = \mathrm{vec}(\mathbf{X})$. To simplify the expression inside the vectorization operator, let $\mathbf{Z} = (\mathbf{A} \otimes \mathbf{C})\,\mathbf{X}^T \in \mathbb{R}^{(n_1 n_2) \times n_3}$. This matrix can be calculated using what was described in the previous subsection. Finally, we obtain the following:

$$\mathbf{y} = \mathrm{vec}\left(\mathbf{C}\mathbf{Z}^T\right). \tag{C.2}$$

Again notice that if all matrices are triangular all operations can be performed without additional memory.

C.1.2 Solving Linear Systems

Now let $\mathbf{A} \in \mathbb{R}^{m \times m}$, $\mathbf{B} \in \mathbb{R}^{n \times n}$ and $\mathbf{y} \in \mathbb{R}^{mn}$. We wish to solve the linear system:

$$(\mathbf{A} \otimes \mathbf{B})\,\mathbf{x} = \mathbf{y},$$

for $\mathbf{x} \in \mathbb{R}^{mn}$. Let $\mathbf{X} \in \mathbb{R}^{m \times n}$ and $\mathbf{Y} \in \mathbb{R}^{m \times n}$ be such that $\mathbf{x} = \mathrm{vec}(\mathbf{X})$ and $\mathbf{y} = \mathrm{vec}(\mathbf{Y})$, respectively. Using, again, the properties of the vectorization operator, we obtain:

$$\mathbf{B}\mathbf{X}\mathbf{A}^T = \mathbf{Y}.$$

Therefore, we can find $\mathbf{X}$ by solving two linear systems:

$$\mathbf{B}\mathbf{Z} = \mathbf{Y}, \tag{C.3}$$

$$\mathbf{A}\mathbf{X}^T = \mathbf{Z}^T. \tag{C.4}$$

If $\mathbf{A}$ and $\mathbf{B}$ are triangular matrices, then no additional memory is required.

Finally, let $\mathbf{C} \in \mathbb{R}^{s\times s}$ be another matrix and $\mathbf{y} \in \mathbb{R}^{nms}$. We wish to solve the linear system:

$$(\mathbf{A} \otimes \mathbf{B} \otimes \mathbf{C})\,\mathbf{x} = \mathbf{y},$$

for $\mathbf{x} \in \mathbb{R}^{nms}$. Let $\mathbf{X} \in \mathbb{R}^{s\times(mn)}$ and $\mathbf{Y} \in \mathbb{R}^{s\times(mn)}$ be such that $\mathbf{x} = \mathrm{vec}(\mathbf{X})$ and $\mathbf{y} = \mathrm{vec}(\mathbf{Y})$, respectively. Then

$$\mathbf{C}\mathbf{X}\,(\mathbf{A} \otimes \mathbf{B})^T = \mathbf{Y}.$$

We start by solving the system:

$$\mathbf{C}\mathbf{Z} = \mathbf{Y},$$

and then solve the system:

$$(\mathbf{A} \otimes \mathbf{B})\,\mathbf{X}^T = \mathbf{Z}^T,$$

using the results of the previous paragraph on each of the rows of $\mathbf{X}$ and $\mathbf{Z}$.

C.2 Implementation Details

Given a set of hyper-parameters $\boldsymbol{\theta}$, the various statistics may be evaluated efficiently in the following sequence:

1. Compute the Cholesky decomposition of all covariance matrices:

$$\mathbf{A}_\xi = \mathbf{L}_\xi \mathbf{L}_\xi^T, \tag{C.5}$$

$$\mathbf{A}_s = \mathbf{L}_s \mathbf{L}_s^T, \tag{C.6}$$

$$\mathbf{A}_t = \mathbf{L}_t \mathbf{L}_t^T. \tag{C.7}$$

2. Scale the outputs by solving in place the linear system:

$$\left(\mathbf{L}_{\xi} \otimes \mathbf{L}_{s} \otimes \mathbf{L}_{t}\right) \tilde{\mathbf{Y}} = \mathbf{Y}. \tag{C.8}$$

3. Scale the design matrices by solving in place the linear systems:

$$\mathbf{L}_{\xi} \tilde{\mathbf{H}}_{\xi} = \mathbf{H}_{\xi}, \tag{C.9}$$

$$\mathbf{L}_{s} \tilde{\mathbf{H}}_{s} = \mathbf{H}_{s}, \tag{C.10}$$

$$\mathbf{L}_{t} \tilde{\mathbf{H}}_{t} = \mathbf{H}_{t}. \tag{C.11}$$

4. Calculate the QR factorizations of the scaled design matrices:

$$\tilde{\mathbf{H}}_{\xi} = \mathbf{Q}_{\xi} \mathbf{R}_{\xi}, \mathbf{Q}_{\xi} = \left[\mathbf{Q}_{\xi,1} \ \mathbf{Q}_{\xi_2}\right], \mathbf{R}_{\xi} = \left[\mathbf{R}_{\xi,1}^{T} \ \mathbf{0}\right]^{T}, \tag{C.12}$$

$$\tilde{\mathbf{H}}_{s} = \mathbf{Q}_{s} \mathbf{R}_{s}, \mathbf{Q}_{s} = \left[\mathbf{Q}_{s,1} \ \mathbf{Q}_{s_2}\right], \mathbf{R}_{s} = \left[\mathbf{R}_{s,1}^{T} \ \mathbf{0}\right]^{T}, \tag{C.13}$$

$$\tilde{\mathbf{H}}_{t} = \mathbf{Q}_{t} \mathbf{R}_{t}, \mathbf{Q}_{t} = \left[\mathbf{Q}_{t,1} \ \mathbf{Q}_{t_2}\right], \mathbf{R}_{t} = \left[\mathbf{R}_{t,1}^{T} \ \mathbf{0}\right]^{T}, \tag{C.14}$$

$$\tag{C.15}$$

where for $\alpha = \boldsymbol{\xi}, s, t$, $\mathbf{Q}_{\alpha,1} \in \mathbb{R}^{n_\alpha \times m_\alpha}, \mathbf{Q}_{\alpha,2} \in \mathbb{R}^{n_\alpha \times (n_\alpha - m_\alpha)}$ and $\mathbf{R}_{\alpha,1} \in \mathbb{R}^{m_\alpha \times m_\alpha}$ is upper triangular.

5. Find $\hat{\mathbf{B}}$ by solving in place the upper triangular system:

$$\left(\mathbf{R}_{\xi,1} \otimes \mathbf{R}_{s,1} \otimes \mathbf{R}_{t,1}\right) \hat{\mathbf{B}} = \left(\mathbf{Q}_{\xi,1} \otimes \mathbf{Q}_{s,1} \otimes \mathbf{Q}_{t,1}\right) \tilde{\mathbf{Y}}. \tag{C.16}$$

6. Calculate (by doing n rank-1 updates) $\hat{\boldsymbol{\Sigma}}$:

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{n-m} \left[\tilde{\mathbf{Y}} - \left(\tilde{\mathbf{H}}_{\xi} \otimes \tilde{\mathbf{H}}_{s} \otimes \tilde{\mathbf{H}}_{t}\right) \hat{\mathbf{B}}\right]^{T} \left[\tilde{\mathbf{Y}} - \left(\tilde{\mathbf{H}}_{\xi} \otimes \tilde{\mathbf{H}}_{s} \otimes \tilde{\mathbf{H}}_{t}\right) \hat{\mathbf{B}}\right]. \tag{C.17}$$

7. Calculate the Cholesky decomposition of $\hat{\boldsymbol{\Sigma}}$:

$$\hat{\boldsymbol{\Sigma}} = \mathbf{L}_{\boldsymbol{\Sigma}} \mathbf{L}_{\boldsymbol{\Sigma}}^{T}. \tag{C.18}$$

8. Now we can evaluate all the determinants involved in the posterior of $\boldsymbol{\theta}$:

$$
\begin{aligned}
\log|\mathbf{A}_\xi| &= 2\sum_{i=1}^{n_\xi}\log L_{\xi,ii}, & \text{(C.19)}\\
\log|\mathbf{A}_s| &= 2\sum_{i=1}^{n_s}\log L_{s,ii}, & \text{(C.20)}\\
\log|\mathbf{A}_t| &= 2\sum_{i=1}^{n_t}\log L_{t,ii}, & \text{(C.21)}\\
\log|\mathbf{H}_\xi^T\mathbf{A}_\xi^{-1}\mathbf{H}_\xi| &= 2\sum_{i=1}^{m_\xi}\log|R_{\xi,1,ii}|, & \text{(C.22)}\\
\log|\mathbf{H}_s^T\mathbf{A}_s^{-1}\mathbf{H}_s| &= 2\sum_{i=1}^{m_s}\log|R_{s,1,ii}|, & \text{(C.23)}\\
\log|\mathbf{H}_t^T\mathbf{A}_t^{-1}\mathbf{H}_t| &= 2\sum_{i=1}^{m_t}\log|R_{t,1,ii}|, & \text{(C.24)}\\
\log|\hat{\boldsymbol{\Sigma}}| &= 2\sum_{i=1}^{q}\log L_{\boldsymbol{\Sigma},ii}, & \text{(C.25)}\\
\log|\mathbf{A}| &= \frac{n}{n_\xi}\log|\mathbf{A}_\xi| + \frac{n}{n_s}\log|\mathbf{A}_s| + \frac{n}{n_t}\log|\mathbf{A}_t|, & \text{(C.26)}\\
\log|\mathbf{H}^T\mathbf{A}^{-1}\mathbf{H}| &= \frac{m}{m_\xi}\log|\mathbf{H}_\xi^T\mathbf{A}_\xi^{-1}\mathbf{H}_\xi| + & \text{(C.27)}\\
&\quad \frac{m}{m_s}\log|\mathbf{H}_s^T\mathbf{A}_s^{-1}\mathbf{H}_s| + & \text{(C.28)}\\
&\quad \frac{m}{m_t}\log|\mathbf{H}_t^T\mathbf{A}_t^{-1}\mathbf{H}_t|. & \text{(C.29)}
\end{aligned}
$$

C.3 Fast Cholesky Updates

This section is concerned with updating the Cholesky decomposition of a covariance matrix in $O(n^2)$ time when a new data point is observed. They are useful in two occasions:

1. When we are doing active learning without updating the hyperparameters.

2. When we wish to sample sequentially the joint distribution in order to obtain a response surface (see Section **??**).

Let $\mathbf{A}_n \in \mathbb{R}^{n\times n}$ be a symmetric positive definite matrix and assume that we have already calculated its Cholesky decomposition $\mathbf{L}_n \in \mathbb{R}^{n\times n}$ (lower triangular). Now let $\mathbf{A}_{n+m} \in R^{(n+m)\times(n+m)}$ be another symmetric positive definite matrix (e.g. the one we obtain if we observe m new data points). In particular let it be given by:

$$\mathbf{A}_{n+m} = \begin{pmatrix} \mathbf{A}_n & \mathbf{B} \\ \mathbf{B}^T & \mathbf{C} \end{pmatrix},$$

where $\mathbf{B} \in \mathbb{R}^{n\times m}$ (e.g. cross covariance) and $\mathbf{C} \in \mathbb{R}^{m\times m}$ (e.g. covariance matrix of the new data). Let $\mathbf{L}_{n+m} \in \mathbb{R}^{(n+m)\times(n+m)}$ be the lower triangular Cholesky factor of $\mathbf{A}_{n+m}$ (i.e. $\mathbf{A}_{n+m} = \mathbf{L}_{n+m}\mathbf{L}_{n+m}^T$). It is convenient to represent it in the matrix block form:

$$\mathbf{L}_{n+m} = \begin{pmatrix} \mathbf{D}_{11} & \mathbf{0}_{n\times m} \\ \mathbf{D}_{21} & \mathbf{D}_{22} \end{pmatrix},$$

where $\mathbf{D}_{11} \in \mathbb{R}^{n\times n}, \mathbf{D}_{21} \in \mathbb{R}^{m\times n}$ and $\mathbf{D}_{22} \in \mathbb{R}^{m\times m}$. We will derive formulas for the efficient calculation of the $\mathbf{D}_{ij}$ based on the Cholesky decomposition of $\mathbf{A}_n$. Notice that:

$$\begin{aligned}
\mathbf{A}_{n+m} &= \mathbf{L}_{n+1}\mathbf{L}_{n+1}^T \\
\Rightarrow \begin{pmatrix} \mathbf{A}_n & \mathbf{B} \\ \mathbf{B}^T & \mathbf{C} \end{pmatrix} &= \begin{pmatrix} \mathbf{D}_{11} & \mathbf{0}_{n\times m} \\ \mathbf{D}_{21} & \mathbf{D}_{22} \end{pmatrix} \begin{pmatrix} \mathbf{D}_{11} & \mathbf{0}_{n\times m} \\ \mathbf{D}_{21} & \mathbf{D}_{22} \end{pmatrix}^T \\
\Rightarrow \begin{pmatrix} \mathbf{L}_n\mathbf{L}_n^T & \mathbf{B} \\ \mathbf{B}^T & \mathbf{C} \end{pmatrix} &= \begin{pmatrix} \mathbf{D}_{11}\mathbf{D}_{11}^T & \mathbf{D}_{11}\mathbf{D}_{21}^T \\ \mathbf{D}_{21}\mathbf{D}_{11}^T & \mathbf{D}_{21}\mathbf{D}_{21}^T + \mathbf{D}_{22}\mathbf{D}_{22}^T \end{pmatrix}.
\end{aligned}$$

From the above equation, we see right away that $\mathbf{D}_{11} = \mathbf{L}_n$. $\mathbf{D}_{21}$ can be found by solving the triangular system $\mathbf{L}_n\mathbf{D}_{21}^T = \mathbf{B}$ and finally $\mathbf{D}_{22}$ is the lower triangular

Cholesky factor of $\mathbf{C} - \mathbf{D}_{21}\mathbf{D}_{21}^T$. To wrap it up, here is how the update should be performed:

1. Set $\mathbf{D}_{11} = \mathbf{L}_n$.
2. Solve the following system for $\mathbf{D}_{21}$:

$$\mathbf{L}_n\mathbf{D}_{21}^T = \mathbf{B}.$$

3. Compute the Cholesky decomposition of:

$$\mathbf{D}_{22}\mathbf{D}_{22}^T = \mathbf{C} - \mathbf{D}_{21}\mathbf{D}_{21}^T,$$

   to find $\mathbf{D}_{22}$.

For the special (but common in practice) case where $m = 1$, then $\mathbf{D}_{21}$ is a vector and $\mathbf{C}$ and $\mathbf{D}_{22}$ are numbers, step 3 can be replaced by:

$$\mathbf{D}_{22} = \sqrt{\mathbf{C} - \mathbf{D}_{21}\mathbf{D}_{21}^T}.$$

We can also update solutions of linear systems. Suppose that we have already solved the following system:

$$\mathbf{L}_n\mathbf{x}_n = \mathbf{y}_n,$$

and after observing m more data points, we need to solve the following:

$$\mathbf{L}_{n+m}\mathbf{x}_{n+m} = \mathbf{y}_{n+m}.$$

If you let

$$\mathbf{y}_{n+m} = \begin{pmatrix} \mathbf{y}_n \\ \mathbf{z} \end{pmatrix},$$

it is trivial to show that $\mathbf{x}_{n+m}$ is given by:

$$\mathbf{x}_{n+m} = \begin{pmatrix} \mathbf{x}_n \\ \mathbf{D}_{22}^{-1}\left(\mathbf{z} - \mathbf{D}_{21}\mathbf{x}_n\right) \end{pmatrix}.$$

BIBLIOGRAPHY

[1] Jørg E. Aarnes, Vegard Kippe, Knut-Andreas Lie, and Alf B. Rustad. Modelling of Multiscale Structures in Flow Simulations for Petroleum Reservoirs. In Geir Hasle, Knut-Andreas Lie, and Ewald Quak, editors, *Geometric Modelling, Numerical Simulation, and Optimization*, chapter 10, pages 307–360. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[2] M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, volume 55. Dover Publications, 1964.

[3] I. Babuška, F. Nobile, and R. Tempone. A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM Journal of Numerical Analysis*, 45(3):1005–1034, 2007.

[4] Ilias Bilionis and Nicholas Zabaras. Multi-output local gaussian process regression: Applications to uncertainty quantification. *Journal of Computational Physics*, 231(17):5718 – 5746, 2012.

[5] Ilias Bilionis and Nicholas Zabaras. Multidimensional adaptive relevance vector machines for uncertainty quantification. *SIAM Journal on Scientific Computing*, 34(6):B881–B908, 2012.

[6] Ilias Bilionis, Nicholas Zabaras, Bledar A. Konomi, and Guang Lin. Multi-output separable gaussian process: Towards an efficient, fully bayesian paradigm for uncertainty quantification. *Journal of Computational Physics*, 241(0):212 – 239, 2013.

[7] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[8] P. Bochev and R. B. Lehoucq. On finite element solution of the pure neumann problem. *SIAM Rev*, 47:50–66, 2001.

[9] Gilles Bourgault and Denis Marcotte. Multivariable variogram and its application to the linear model of coregionalization. *Mathematical Geology*, 23:899–928, 1991.

[10] Adrian W. Bowman and Adelchi Azzalini. *Applied Smoothing Techniques for Data Analysis: The Kernel Approach with S-Plus Illustrations (Oxford Statistical Science Series)*. Oxford University Press, USA, November 1997.

[11] P. Boyle and M. Frean. Dependent gaussian processes. In *Neural Information Processing Systems 18*, pages 217–224, 2005.

[12] F. Brezzi, T. J. R. Hughes, L. D. Marini, and A. Masud. Mixed discontinuous galerkin methods for darcy flow. *J. Sci. Comput.*, 22-23(1):119–145, June 2005.

[13] George Casella and Edward I. George. Explaining the Gibbs sampler. *The American Statistician*, 46(3):167–174, 1992.

[14] K. Chaloner and I. Verdinelli. Bayesian Experimental Design : A Review. *Statistical Science*, 10(3):273–304, 1995.

[15] Z. Chen and T. Y. Hou. A mixed multiscale finite element method for elliptic problems with oscillating coefficients. *Mathematics of Computation*, 72(242):541–576, 2003.

[16] H. A. Chipman, E. I. George, and R. E. McCulloch. Bayesian CART model search. *Journal of the American Statistical Association*, 93(443):935–948, September 1998.

[17] H. A. Chipman, E. I. George, and R. E. McCulloch. Bayesian treed models. *Machine Learning*, 48(1):299–320, 2002.

[18] D. A. Cohn. Neural network exploration using optimal experiment design. *Neural Networks*, 9(6), August 1996.

[19] Stefano Conti and Anthony OHagan. Bayesian emulation of complex multi-output and dynamic computer models. *Journal of Statistical Planning and Inference*, 140(3):640 – 651, 2010.

[20] Corinna Cortes and Vladimir Vapnik. Support-vector networks. In *Machine Learning*, pages 273–297, 1995.

[21] N. A. C. Cressie. *Statistics for Spatial Data*, volume 4. Wiley, New York, 1991.

[22] C. Currin, T. Mitchell, M. Morris, and D. Ylvisaker. A Bayesian approach to the design and analysis of computer experiments. Technical Report ORNL–6498, Oak Ridge Laboratory, 1988.

[23] Carla Currin, Toby Mitchell, Max Morris, and Don Ylvisaker. Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments. *Journal of the American Statistical Association*, 86(416):953–963, 1991.

[24] A. P. Dawid. Some matrix-variate distribution theory: Notational considerations and a Bayesian application. *Biometrika*, 68(1):265–274, 1981.

[25] S. De Iaco, D. Myers, and D. Posa. The linear coregionalization model and the productsum spacetime variogram. *Mathematical Geology*, 35:25–38, 2003.

[26] Daniele A. Di Pietro, Stefania Lo Forte, and Nicola Parolini. Mass preserving finite element implementations of the level set method. *Applied Numerical Mathematics*, 56(9):1179–1195, September 2006.

[27] R. Fletcher. *Practical Methods of Optimization*. Wiley, 2nd edition, 1987.

[28] J. Foo, X. Wan, and G. E. Karniadakis. The Multi-Element probabilistic collocation method (ME-PCM): error analysis and applications. *Journal of Computational Physics*, 227(22):9572–9595, 2008.

[29] D. Francois, V. Wertz, and M. Verleysen. About the locality of kernels in high-dimensional spaces. In *International Symposium on Applied Stochastic Models and Data Analysis*, pages 238–245, Brest (France), 2005.

[30] M. Galassi, J. Davies, J. Theiler, B. Gough, G. Jungman, P. Alken, M. Booth, and F. Rossi. *GNU Scientific Library Reference Manual*. 2009.

[31] B. Ganapathysubramanian and N. Zabaras. A stochastic multiscale framework for modeling flow through heterogeneous porous media. *Journal of Computational Physics*, 228(2):591–618, 2009.

[32] Walter Gautschi. Algorithm 726: ORTHPOL-a package of routines for generating orthogonal polynomials and Gauss-type quadrature rules. *ACM Transactions on Mathematical Software*, 20(1):21–62, March 1994.

[33] R. G. Ghanem and P. D. Spanos. *Stochastic Finite Elements: A Spectral Approach*. Dover Publications, 2003.

[34] Roger G. Ghanem and Pol D. Spanos. *Stochastic finite elements: a spectral approach*. Springer-Verlag, New York, 1991.

[35] M. B. Giles. Multilevel Monte Carlo path simulation. Technical report, Oxford University Computing Laboratory, 2006.

[36] M. B. Giles. Improved multilevel Monte Carlo convergence using the Milstein scheme. In A. Keller, S. Heinrich, and H. Niederreiter, editors, *Monte Carlo and Quasi-Monte Carlo Methods 2006*, pages 343–358. Springer, 2008.

[37] Gene H. Golub and Charles F. Van Loan. *Matrix Computations (Johns Hopkins Studies in Mathematical Sciences)(3rd Edition)*. The Johns Hopkins University Press, 3rd edition, October 1996.

[38] M. Goulard and M. Voltz. Linear coregionalization model: Tools for estimation and choice of cross-variogram matrix. *Mathematical Geology*, 24:269–286, 1992.

[39] Robert B. Gramacy and Herbert K. Lee. Cases for the nugget in modeling computer experiments. *Statistics and Computing*, 22(3):713–722, May 2012.

[40] Robert B. Gramacy and Herbert K. Lee. Cases for the nugget in modeling computer experiments. *Statistics and Computing*, 22(3):713–722, May 2012.

[41] Robert B Gramacy and Herbert K. H Lee. Bayesian treed Gaussian Process models with an application to computer modeling. *Journal of the American Statistical Association*, 103(483):1119–1130, September 2008.

[42] Robert B. Gramacy and Herbert K. H. Lee. Adaptive design and analysis of supercomputer experiments. *Technometrics*, 51(2):130–145, May 2009.

[43] Robert B. Gramacy and Herbert K. H. Lee. Cases for the nugget in modeling computer experiments. *Statistics and Computing*, (to appear), December 2011.

[44] W. W. Hager. Updating the inverse of a matrix. *SIAM Review*, 31(2):221–239, 1989.

[45] Helmut Harbrecht, Michael Peters, and Reinhold Schneider. On the low-rank approximation by the pivoted cholesky decomposition. *Appl. Numer. Math.*, 62(4):428–440, April 2012.

[46] D. A. Harville. *Matrix Algebra From a Statisticians Perspective*. Springer-Verlag, 1997.

[47] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, April 1970.

[48] M. A. Heroux and J. M. Willenbring. Trilinos Users Guide. Technical report, Sandia National Laboratories, 2003.

[49] Michael A. Heroux, Roscoe A. Bartlett, Vicki E. Howle, Robert J. Hoekstra, Jonathan J. Hu, Tamara G. Kolda, Richard B. Lehoucq, Kevin R. Long, Roger P. Pawlowski, Eric T. Phipps, Andrew G. Salinger, Heidi K. Thornquist, Ray S. Tuminaro, James M. Willenbring, Alan Williams, and Kendall S. Stanley. An overview of the trilinos project. *ACM Trans. Math. Softw.*, 31(3):397–423, 2005.

[50] Dave Higdon, James Gattiker, Brian Williams, and Maria Rightley. Computer model calibration using high-dimensional output. *Journal of the American Statistical Association*, 103(482):570–583, 2008.

[51] R. L. Iman and W. J. Conover. Small sample sensitivity analysis techniques for computer models, with an application to risk assessment. *Communications in Statistics - Theory and Methods*, 9(17):1749–1842, 1980.

[52] Marc C. Kennedy and Anthony O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):425–464, 2001.

[53] V. Kippe, J. E. Aarnes, and K. A. Lie. A comparison of multiscale methods for elliptic problems in porous media flow. *Computational Geosciences*, 12:377–398, 2008.

[54] G. Lin and A. Tartakovsky. An efficient, high-order probabilistic collocation method on sparse grids for three-dimensional flow and solute transport in randomly heterogeneous porous media. *Advances in Water Resources*, 32(5):712–722, 2009.

[55] G. Lin and A. Tartakovsky. Numerical studies of three-dimensional stochastic Darcys equation and stochastic advection–diffusion–dispersion equation. *Journal of Scientific Computing*, 43(1):92–117, 2010.

[56] J. S. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer, 2008.

[57] A. Logg, K.-A. Mardal, and G. N. Wells, editors. *Automated Solution of Dif-*

ferential Equations by the Finite Element Method, volume 84 of *Lecture Notes in Computational Science and Engineering*. Springer, 2012.

[58] Anders Logg, Garth N. Wells, and Johan Hake. *DOLFIN: a C++/Python Finite Element Library*, chapter 10. Springer, 2012.

[59] X. Ma and N. Zabaras. An adaptive hierarchical sparse grid collocation algorithm for the solution of stochastic differential equations. *Journal of Computational Physics*, 228(8):3084–3113, 2009.

[60] X. Ma and N. Zabaras. A stochastic mixed finite element heterogeneous multiscale method for flow in porous media. *Journal of Computational Physics*, 230(12):4696–4722, 2011.

[61] D. J. C. MacKay. Introduction to Gaussian processes. In C. M. Bishop, editor, *Neural Networks and Machine Learning*, volume 168, pages 133–166, 1998.

[62] David J.C. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4:590–604, 1992.

[63] J.R. Magnus and H. Neudecker. *Matrix Differential Calculus With Applications in Statistics and Econometrics*. Wiley Series in Probability and Statistics. John Wiley, 1999.

[64] M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):pp. 239–245, 1979.

[65] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[66] C. A. Micchelli and M. Pontil. Kernels for multi-task learning. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing 17*, pages 921–928. MIT Press, 2005.

[67] R. M. Neal. *Bayesian Learning for Neural Networks*. Springer, New York, 1996.

[68] R. M. Neal. Monte Carlo implementation of Gaussian process models for

bayesian regression and classification. Technical Report 9702, Departement of Statistics, University of Toronto, Toronto, 1997.

[69] F. Nobile, R. Tempone, and C. Webster. A sparse grid collocation method for elliptic partial differential equations with random input data. *SIAM Journal of Numerical Analysis*, 45:2309–2345, 2008.

[70] F. Nobile, R. Tempone, and C. G. Webster. A sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM Journal of Numerical Analysis*, 46(5):2309–2345, 2008.

[71] J Oakley and A O'Hagan. Bayesian inference for the uncertainty distribution of computer model outputs. *Biometrika*, 89(4):769–784, 2002.

[72] A. O'Hagan. Monte carlo is fundamentally unsound. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 36(2/3):pp. 247–249, 1987.

[73] A. O'Hagan. Bayes-Hermite Quadrature. *Journal of Statistical Planning and Inference*, 29:245–260, 1991.

[74] Anthony O'Hagan, J. M. Bernardo, J. O. Berger, A. P. Dawid, A. F. M. Smith (eds, Marc C. Kennedy, and Jeremy E. Oakley. Uncertainty analysis and other inference tools for complex computer codes, 1998.

[75] R. Piessens, E. de Doncker-Kapenga, C. W. Ueberhuber, and D. K. Kahaner. *QUADPACK: A subroutine package for automatic integration*. Springer Verlag, 1983.

[76] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, 3rd edition, 2007.

[77] C. E. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[78] Carl Edward Rasmussen and Zoubin Ghahramani. Bayesian monte carlo. In *In Proceedings of NIPS 15*, pages 489–496. MIT Press, 2003.

[79] P. Raviart and J. Thomas. A mixed finite element method for 2-nd order elliptic problems. In Ilio Galligani and Enrico Magenes, editors, *Mathematical Aspects of Finite Element Methods*, volume 606 of *Lecture Notes in Mathematics*, chapter 19, pages 292–315. Springer Berlin / Heidelberg, 1977.

[80] Murray Rosenblatt. Remarks on a Multivariate Transformation. *The Annals of Mathematical Statistics*, 23:470–472, 1952.

[81] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4(4):409–435, 1989.

[82] T. J. Santer, B. J. Williams, and W. I. Notz. *The Design and Analysis of Computer Experiments*. Springer, New York, 2003.

[83] S. Seo, M. Wallat, T. Graepel, and K. Obermayer. Gaussian process regression: active data selection and test point rejection. In *International Joint Conference on Neural Networks*, volume 3, pages 241–246, Los Alamitos, CA, 2000. IEEE Press.

[84] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall/CRC, London, 1998.

[85] S. A. Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. In *Doklady Akademii Nauk SSSR*, volume 4, page 123, 1963.

[86] Matthew A. Taddy, Robert B. Gramacy, and Nicholas G. Polson. Dynamic trees for learning and design. *Journal of the American Statistical Association*, 106(493):109–123, March 2011.

[87] Y. W. Teh, M. Seeger, and M. I. Jordan. Semiparametric latent factor models. In R. G. Cowell and Z. Ghahramani, editors, *10th International Workshop on Artificial Intelligence and Statistics*, pages 333–340. Society for Artificial Intelligence and Statistics, 2005.

[88] M. E. Tipping. Sparse Bayesian learning and the Relevance Vector Machine. *Journal of Machine Learning Research*, 1:211–245, 2001.

[89] M. E. Tipping and A. Faul. Fast marginal likelihood maximisation for sparse Bayesian models. In *Proceedings of the ninth international workshop on artificial intelligence and statistics*, volume 1, pages 1–13, 2003.

[90] Charles F. van Loan. The ubiquitous Kronecker product. *J. Comput. Appl. Math.*, 123(1-2):85–100, November 2000.

[91] X. Wan and G. E. Karniadakis. An adaptive multi-element generalized

polynomial chaos method for stochastic differential equations. *Journal of Computational Physics*, 209:617–642, 2005.

[92] X. Wan and G. E. Karniadakis. Multi-element generalized polynomial chaos for arbitrary probability measures. *SIAM Journal of Scientific Computing*, 28(3):901–928, 2006.

[93] William J. Welch, Robert J. Buck, Jerome Sacks, Henry P. Wynn, Toby J. Mitchell, and Max D. Morris. Screening, predicting, and computer experiments. *Technometrics*, 34(1):15–25, February 1992.

[94] C. K. I. Williams and C. E. Rasmussen. Gaussian processes for regression. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*, pages 514–520. MIT Press, 1996.

[95] D. Xiu. High-order collocation methods for differential equations with random inputs. *SIAM Journal on Scientific Computing*, 27(3):1118–1139, 2005.

[96] D. Xiu. Efficient collocational approach for parametric uncertainty analysis. *Communications in Computational Physics*, 2(2):293–309, 2007.

[97] D. Xiu. *Numerical Methods for Stochastic Computations: A Spectral Method Approach*. Princeton University Press, 2010.

[98] D. Xiu and J. S. Hesthaven. High-order collocation methods for differential equations with random inputs. *SIAM Journal on Scientific Computing*, 27(3):1118–1139, 2005.

[99] Dongbin Xiu and George Em Karniadakis. The wiener–askey polynomial chaos for stochastic differential equations. *SIAM Journal of Scientific Computing*, 24(2):619–644, February 2002.